2006

Evolutionary multi-objective optimization for gating and riser system design of metal castings

Jean Shang Leen Kor
University of Windsor

Evolutionary Multi-Objective Optimization for Gating and Riser System Design of Metal Castings

by

Jean Shang Leen Kor

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada

2006

Your file Votre référence
ISBN: 978-0-494-34954-0
Our file Notre référence
ISBN: 978-0-494-34954-0

Abstract

The gating and riser system design plays an important role in the quality and cost of a metal casting. Due to the lack of existing theoretical procedures to follow, the design process is carried out on a trial-and-error basis. The casting design optimization problem is characterized by multiple design variables, conflicting objectives, and a complex search space, making it unsuitable for sensitivity-based optimization.

In this study, a formal optimization method using evolutionary techniques was developed to overcome such complexities. A framework for integrating the optimization procedure with numerical simulation for the design evaluation is presented. The comparison between a scalar and vector optimization approach was explored using the weighted-sum and multi-objective Genetic Algorithm methods. The proposed optimization framework was applied to the gating and riser system of a sand casting and the results were compared to a popular Design-of-Experiment (DOE) method. It showed that the multi-objective method gave better results and provided more flexibility in decision making.

Acknowledgements

First of all, I would like to express my sincere gratitude to my advisors Dr. Xiang Chen and Dr. Henry Hu for giving me the opportunity to become involved in this research. Their continuous suggestions, guidance, and support has not only helped me achieve my thesis goals but also helped me grow as a better researcher. I would like to thank my industrial advisor, Dr Qigui Wang from General Motors Powertrain for his valuable suggestions and advice.

My warmest thanks also go to Dr. Behnam Shahrrava, the department reader, and Dr. Randy Bowers, the outside program reader for their constructive feedback and advice towards the fulfillment of this thesis work.

This research would not have been possible without the help of Mr. Soren Anderson of MAGMASOFT GmbH and I would like to thank him for lending his technical expertise. I would also like to acknowledge Kanpur Genetic Algorithms Laboratory for providing the NSGA-II code.

I would also like to thank the Department of Electrical and Computer Engineering for the support of my graduate studies in these two years as well as my fellow graduate students in the department for their inspiring discussions and support. Funding for the course of this research has been provided by General Motors and is also gratefully acknowledged.

Contents

List of Figures

List of Tables

List of Abbreviations

CFD	Computational Fluid Dynamics
DOE	Design of Experiments
EA	Evolutionary Algorithm
GA	Genetic Algorithm
MOEA	Multi-Objective Evolutionary Algorithm
NSGA II	Non-Dominated Sorting Genetic Algorithm II
OLH	Optimal Latin Hypercube
WS-GA	Weighted-Sum Genetic Algorithm

Chapter 1
Introduction

1.1 Casting Design Background

Casting is a process by which molten metal is introduced into a mold and allowed to solidify to form the desired shaped product. Due to its ability to form complex shapes easily and economically in large quantities, the casting process is particularly applied in mass production. Although the origins of this process dates back to many centuries ago, it is still widely used today especially in the automotive manufacturing industry [1].

Casting design, in particular the gating and riser system design, has a direct influence on the quality of cast components [2-6]. The gating system is used to introduce metal into the mould cavity whereas risers are used to compensate for the shrinkage of the casting as it solidifies. The design of gating and riser systems is largely based on past experience and empirical rules [7-9] and like most engineering design problems, casting design is done on a trial and error basis. With this approach, finding an acceptable gating and riser system design proves to be an expensive and arduous process.

With the availability of modern numerical software, simulation has become an important tool for the design, analysis and optimization of casting processes [10-15]. Numerical simulation of the casting process provides a powerful means of analyzing various phenomena occurring during casting processes. It can give the designer an insight into the details of fluid flow, heat transfer and solidification as well as a prediction of porosity, inclusions, hot tears, and other casting defects. This allows flexibility for designer to explore different options and helps avoid costly prototype trials. The benefits of computer simulation have been demonstrated not only in the gating and riser design but also in process selection and shape design.

Although recent advances in computer technology have boosted the applications of numerical simulation in various engineering design fields, the design optimization is nonetheless an iterative process where the designer has to do multiple revisions based on personal experience. This process requires intensive human interaction and numerous trial-and-error adjustments as demonstrated in Figure 1-1. Due to the variation of individual knowledge and experience, the design process is often inconsistent.

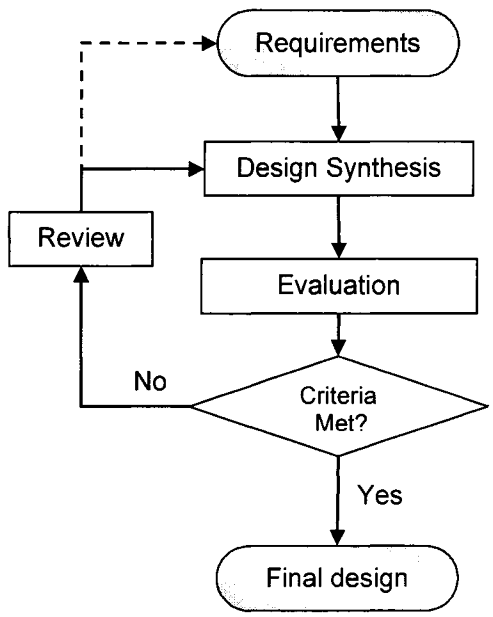


Figure 1-1 The basic design cycle.

In addition, engineering design problems often consist of multiple conflicting objectives that have to be taken into consideration, and the designer is faced with the problem of finding a compromise between them. Even with the assistance of simulation tools, it is difficult to determine the optimal shapes, sizes and locations of the casting components while simultaneously trying to adhere to conflicting quality and cost constraints. Numerous efforts have been made to improve the quality of castings and reduce cost and lead time using simulation tools. Since there is no methodical way of doing it, this process of finding an optimum design proves to be a challenging task.

1.2 Motivation and Challenges

In effort to gain competitive edge, there is an adamant need for increased efficiency and quality in the design process. With apt computer technology readily available, exploiting its power to automate the casting design optimization process makes both economic and engineering sense.

Coupling numerical simulation with formal optimization methods is one way of adopting a more systematic approach towards casting design [1, 16]. Ultimately, the overall design process can be made more efficient; repetitious tasks can be handled methodically, not only considerably cutting down the design cycle time but also delivering optimal products.

The prospect of automating the design optimization process is undoubtedly an attractive vision. Although optimization methods have been successfully applied in many engineering design applications, the application for casting design still lacks sufficient investigation in the public domain. This trend can be largely associated with the nature of casting design, which hinders attempts at formal optimization processes.

First, the casting design practice is particularly impervious to changes due to its history of tradition. It has long been practiced on the basis of experimental trial and error due to the lack of fixed theoretical procedures to follow. Next, the casting design optimization problem is characterized by multiple control points and multiple conflicting objectives that involve many parameters. The problem is exacerbated by a complex and multimodal search space, which often involves objective and constraint functions that are nonlinear, discontinuous and not properly defined.

In addition, the casting design evaluation is a computationally intensive, costly, and time consuming process as it involves complex computational fluid dynamics (CFD) and heat transfer calculations. Furthermore, analytic derivatives are unavailable and the search space is poorly understood, making sensitivity-based optimization methods unsuitable.

The fundamental goal of this thesis is to improve the efficiency of the casting design process by employing simulation and optimization techniques to (1) automate the design process and (2) improve the design quality at the same time. Therefore the first aim of this thesis is to formalize the casting design process and present a framework suitable for the casting design optimization. The optimization framework should be generalized so that extensive parameter tuning is not required for each different casting problem. The focus is primarily on the gating and feeding system where numerical simulation is employed to predict the performance of a design.

The second aim is to develop a reliable multi-objective evolutionary algorithm that not only can handle the complexity of the search space and other aforementioned difficulties of the casting design problem, but also allows flexibility in decision making. The optimization algorithm should be robust regardless of the choice of initial design(s). Since variations in individual knowledge and experience often lead to inconsistent designs, the third aim is to reduce or eliminate context-dependent user settings that can affect the optimization results or introduce further inconsistencies to the results.

1.3 Literature Survey

The design of gating and riser systems has a major impact on the quality of the castings. Light metals such as aluminum castings are especially vulnerable to certain defects such as porosity and oxide inclusions if the casting process is not appropriately selected [1, 3]. These defects have significant influence on the mechanical properties of the castings. Campbell developed ten rules for the design and assurance of high quality castings [5, 13]. Among them are four rules that can be directly linked to the design of the gating and riser system:

- Prevent liquid front damage: Maximum meniscus velocity <0.5 m/sec. No top gating.
- Avoid liquid front arrests: Liquid should not stop at any point along the front, progressing only uphill in a continuous, uninterrupted advance.

- No bubble damage: Bubbles of air entrained by the filling system should not pass through the liquid metal into the mold cavity. Design the sprue and runner to fill in one pass. Avoid the use of wells.
- Eliminate shrinkage damage: No feeding uphill. Follow the feeding rules and run an appropriate solidification model.

To achieve an optimum design of a gating and riser system with minimum lead time and maximum yield and casting quality, it is necessary to incorporate John Campbell's casting design rules with computational tools. However, without a formal optimization strategy, the process of finding an optimum design proves to be a challenging task.

There are very few publications that address the formal casting design optimization in the open literature. Among some of the earlier work inspired by the idea of automating the casting design optimization was that of McDavid and Dantzig [12, 17] associated with fluid flow modeling and numerical analysis. They presented their work using design sensitivity analysis coupled together with finite element analysis for a runner system design optimization.

Other related work that employed the design sensitivity analysis approach includes the casting riser optimization carried out by Dantzig et al [1] and Ebrahimi et al [18], in which direct differentiation methods were used to calculate the sensitivities. Their approach was demonstrated on investment castings. Analysis of the design was based on 2-dimensional calculations, and it was apparent that the sensitivity results were greatly affected by the time step size during the solidification process.

Esparza [10] presented a numerical optimization technique based on a gradient-search method for the gating system design of aluminum castings. In his approach, sequential quadratic programming was used, taking into account the mathematical structure of the problem. This approach required preliminary experiments and the performance was affected by the choice of the starting solution and the step size used.

Instead of coupling with numerical simulation, a geometric approach for optimizing riser designs was proposed by Das et al [20]. He pointed out that using a finite element or finite difference based numerical simulation becomes impractical when time is critical. He proposed an optimization scheme based on the riser modulus calculations for providing a quick estimate of the solidification process. In other related work, Guleyupoglu et al [21] and Jacob et al [22] used genetic algorithms with the modulus criterion to optimize metal yield in riser design.

For computationally complex problems such as casting design, the performance landscape of the design space is characterized by one or more of the following: highly nonlinear, multi-modal and non-analytical, or cannot be expressed explicitly in functional form. These characteristics preclude gradient-based methods. Instead, adaptive search approaches such as Simulated Annealing, Genetic Algorithm, and Tabu Search have been shown to be the more reliable optimization methods for finding optimal solutions [27, 28]. Evolutionary algorithms have also been shown to be effective methods for solving multi-objective problems. They have been successfully applied in several CFD applications that involved numerical simulation such as hydraulic system designs [23], aerodynamic shape designs [24] and others. A collection of multi-objective applications can be found in the review articles of Andersson [25] and Coello [26].

1.4 Thesis Outline

The thesis is divided into six main sections. A background of the casting design problem has been introduced in Chapter 1. Chapter 2 introduces the gating and riser system design. In Chapter 3, a multi-objective problem is introduced along with several different optimization approaches. Preliminary theories about Genetic Algorithms (GA) are presented to give general descriptions of the basic concepts and parameters involved. Two approaches, the Weighted-Sum GA and Multi-Objective Evolutionary Algorithm (MOEA), are further presented along with discussions of their advantages and disadvantages. Chapter 4 demonstrates the application of evolutionary algorithms to the optimization of the gating and riser system design. The problem formulation and implementation of an optimization framework are presented. In Chapter 5, simulation results comparing the scalar and vector optimization approaches are presented. Chapter 6 concludes this research and provides possible directions for future work.

Chapter 2

Gating and Riser Design

2.1 Introduction

In sand casting, molten metal is poured into the mold cavity and left to solidify, forming the desired shaped product via a gating and riser delivery system. The gating system is used to introduce metal into the mould cavity whereas risers are used to compensate for the shrinkage of the casting as it solidifies. Figure 2-1 shows the typical arrangement of components in a casting gating and riser system which consist of a pouring basin, down sprue, sprue well, runners, ingates, and risers.

The gating system and risers play different roles in the production of a casting; the gating system controls the metal flow delivered to the casting cavity whereas the risers ensure adequate feeding of metal to the casting during solidification. However, the design of both systems is critical and often taken into consideration concurrently as they both contribute to the quality and cost of castings.

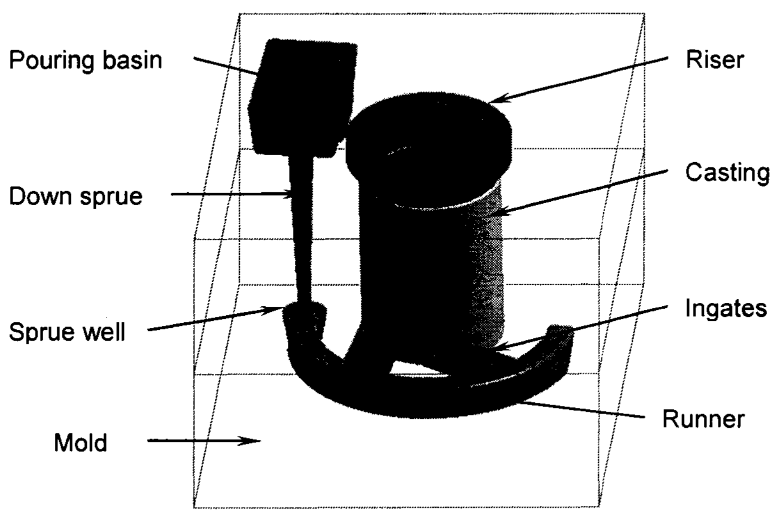


Figure 2-1 Typical elements of gating and riser system design.

2.2 Riser Design

One of the main issues associated with casting solidification is shrinkage. When the solidification front reaches the innermost region or the hot spot, there is no more liquid metal left and shrinkage cavity is formed. To avoid this defect in a casting, risers are designed with the appropriate shapes and sizes to solidify later than the hot spot. Campbell [9] has summarized six main feeding rules for riser design as follows:

1. Heat transfer criterion: The feeder must solidify at the same time or later than the casting.
2. Mass transfer criterion: The feeder must contain sufficient liquid to meet the volume-contraction requirements of the casting.
3. Junction requirement: The junction between the feeder and the casting should not create a hot spot, i.e. be the last to solidify.
4. Feed path requirement: There must be a path to allow feed metal to reach feeding points.
5. Pressure requirement: There must be sufficient pressure at all points in the casting to suppress the formation of cavities.
6. Pressure gradient requirement: There must be sufficient pressure differential requirement to cause the feed material to flow in the right direction.

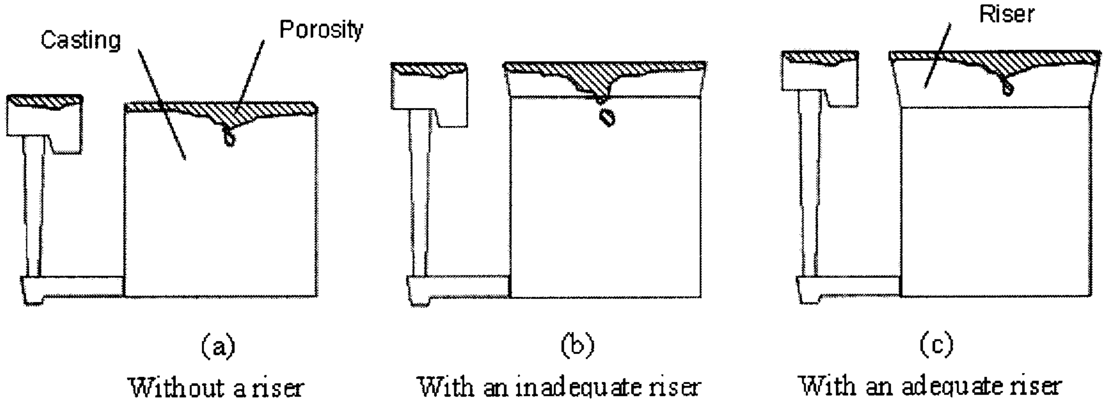


Figure 2-2 The role of risers on the shrinkage porosity formed during the solidification process.

With an adequately sized riser attached to the casting, the shrinkage cavity can be shifted to the riser and cut off after casting solidification as illustrated in Figure 2-2.

Adequate risers are required to avoid solidification related defects such as shrinkage, micro-porosity, hot tears and so on. However, the design of a riser should take into account two conflicting objectives: eliminate shrinkage defects and maximize casting yield. Reducing porosity defects are vital in improving not only the cosmetic quality of a casting but more importantly its strength and functionality. Introducing risers at the correct location and of the correct size is imperative in achieving improved porosity levels. However, in high volume production, material utilization is also a very important aspect which needs to be taken into consideration by minimizing the size of risers.

2.3 Gating Design

In general, the gating system determines how the liquid metal is delivered into the mould cavity. The path of molten metal during filling process comprises of mainly four parts: The pouring of molten metal from the ladle to the pour basin, the metal flow within the sprue to the runners, entry of molten metal from ingate(s) to the mold cavity and the filling of mold cavity. Velocity of the molten metal varies widely within the gating channels as well as inside the mould cavity.

The design of the gating system must satisfy several interacting requirements. Although rapid filling of the mold is desirable, filling speed is restricted by the need to avoid mold erosion and excessive turbulence of the metal flow [17]. Turbulence implies irregular, fluctuating flow with disturbances that are often characterized by high flow velocities of molten metal and obstructed flow paths. When metal is poured rapidly into a mold, the liquid flow is agitated, causing waves to form and resulting in turbulence induced defects [9]. This is illustrated in Figure 2-3. On the other hand, relatively slow filling reduces melt temperature and consequently generates cold shuts within the casting.

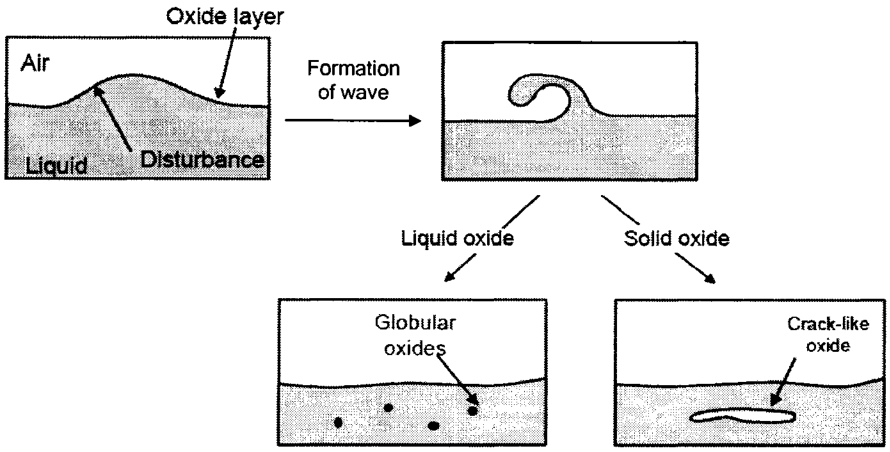


Figure 2-3 Surface turbulence in liquid metal. (adapted from Surface Turbulence in Liquid Metal [29])

A well-designed gating system eases the filling of the mould and minimizes surface turbulence of molten metal, thus minimizing the possibility of solid inclusions, air aspiration and oxide defects. Such defects greatly reduce the fatigue and ultimate strength of a casting. This is of particular importance in aluminum alloy casting which are highly susceptible to oxidation.

Various factors influence the metal flow especially the shapes and sizes of the gating elements. Velocity of molten metal at the ingate mainly depends on the gating ratio A_s: A_r: A_g given by the cross-sectional areas of the sprue exit, runners and ingates respectively. The geometry of the gating components also influences the mold filling patterns. Therefore a properly designed gating system is necessary to control the flow of molten metal, ensuring a smooth and uniform filling pattern and reducing turbulence induced defects.

The term critical velocity, V_{crit} is one way of indicating surface turbulence of the molten metal in a mould [9]. V_{crit} is the limiting condition when a liquid droplet is about to form and can be observed when the inertial pressure of the molten metal is balanced by the surface tension force [29]. This is demonstrated in Figure 2-4.

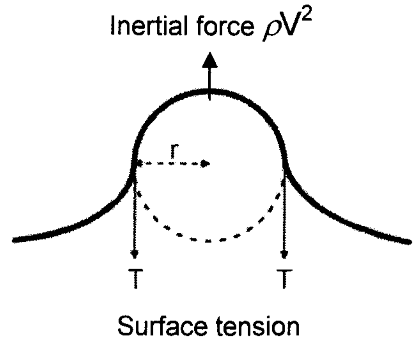


Figure 2-4 Condition for the formation of a droplet. (adapted from Turbulent Flow in Metals [29])

The critical velocity can be defined as:

$$V_{crit} = \sqrt{\frac{2T}{r\rho}}$$

where ρ is the molten metal density, V is the velocity of the disturbance, r is the radius of the droplet shape that is forming and T is the surface tension restraining it. The critical velocity is therefore a simplified indication of surface turbulence in molten metal that can be used as an important quantitative measure in the design of a gating system. For aluminum alloy, the liquid flow should not exceed a velocity of 0.5 m/s to maintain the stability of the meniscus front [30, 31]. Otherwise the surface oxide film may get folded into the bulk of the liquid and constitute initiation sites for gas evolution, shrinkage cavities and hot tears, decreasing leak tightness, corrosion resistance and the strength of the casting.

Chapter 3

Multi-Objective Optimization

3.1 Problem Definition

In a single-objective problem, the idea is to find a set of values for the design variables such that when subject to a number of constraints, yields an optimum value of the single objective or cost function. However, engineering design problems often consist of multiple objectives. In a multi-objective problem, the aim is to find a set of values for the design variables which optimizes a set of objective functions simultaneously. A multi-objective problem with constraints can be defined as:

$$\begin{aligned}&\underset{x \in S}{\text{Min}}\; f_i(\mathbf{x}), \quad i = 1,2,\ldots,k\\&\text{subject to } g_j(\mathbf{x}) \le 0, \quad j = 1,2,\ldots,m\end{aligned}$$

where $S = \{\mathbf{x} \mid g_j(\mathbf{x}) \le 0;\ j = 1,2,\ldots,m\}$ is the feasible design set, x_i, $i = 1,2,..,n$ are decision variables, $f_i(\mathbf{x})$, $i = 1,2,..,k$ are objective functions to be minimized, and $g_j(\mathbf{x})$, $j = 1,2,..,m$ are inequality constraints.

Alternatively, the multi-objective problem can also be represented in the criterion (or objective) space where the axes represent different objective functions. With $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}),\ldots, f_k(\mathbf{x})]^T$, the feasible criterion space $Z = \{\mathbf{f}(\mathbf{x}) \mid \mathbf{x} \in S\}$ is the set of objective function values corresponding to the feasible points in the design space S. Therefore, Z indicates all the points in the criterion space that are obtained using the feasible points in S. Figure 3.1 shows the mapping of a two-dimensional design space S with design variables x_1 and x_2 into the objective space Z with two objectives f_1 and f_2.

The multi-objective optimization problem is also known as vector optimization. Throughout this thesis, the term minimization is used for optimization. A maximization problem $\mathbf{F}(\mathbf{x})$ can be transformed into a minimization problem by $\mathbf{f}(\mathbf{x}) = -\mathbf{F}(\mathbf{x})$ [32].

In a multi-objective problem, minimizing $\mathbf{f}(\mathbf{x})$ lacks clear meaning since functions of different characteristics are incomparable especially when the objectives are inherently conflicting with each other. When the objectives are in conflict with each other, finding a solution that minimizes all of the objectives at the same time becomes impossible. In order to choose the best solution, objective trade-off and subjective judgment from the decision maker are required. Hence the concept of Pareto optimality and dominance is predominantly used for defining solutions in multi-objective optimization problems.

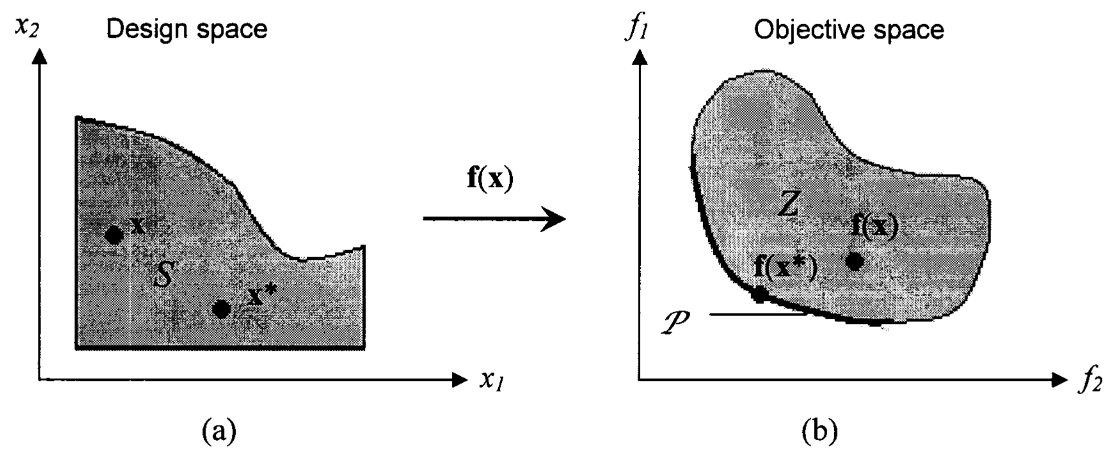


Figure 3-1 (a) A 2-D design space and (b) the mapped 2-D objective space.

Pareto Optimality: A point $\mathbf{x}^* \in S$ is called Pareto optimal if there exists no other point $\mathbf{x} \in S$ such that $\forall i \in \{1,2,\ldots,k\}: f_i(\mathbf{x}) \le f_i(\mathbf{x}^*)$ and $\exists i \in \{1,2,\ldots,k\}: f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$. In other words, a feasible solution $\mathbf{x}^*$ is called Pareto optimal if there is no other feasible solution $\mathbf{x}$ that reduces some objective function without causing a simultaneous increase in at least one other objective function.

Pareto Dominance: The concept on Pareto dominance is similar but refers to the points in the criterion space Z. Considering the objective vector function at two different decision vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ with $\mathbf{f}(\mathbf{x}_a)$, $\mathbf{f}(\mathbf{x}_b) \in Z$, $\mathbf{f}(\mathbf{x}_a)$ is said to dominate $\mathbf{f}(\mathbf{x}_b)$, denoted as

$\mathbf{f}(\mathbf{x}_a) \succ \mathbf{f}(\mathbf{x}_b)$ if $\forall i \in \{1,2,...,k\} : f_i(\mathbf{x}_a) \le f_i(\mathbf{x}_b)$ and $\exists i \in \{1,2,...,k\} : f_i(\mathbf{x}_a) < f_i(\mathbf{x}_b)$. This means no component of $\mathbf{f}(\mathbf{x}_a)$ is larger than the corresponding component of $\mathbf{f}(\mathbf{x}_b)$ and at least one component is smaller. Otherwise, $\mathbf{f}(\mathbf{x}_a)$ is dominated by $\mathbf{f}(\mathbf{x}_b)$.

Accordingly, it can also be said that a point $\mathbf{x}^*$ is Pareto optimal if and only if there is no other point $\mathbf{x}$ for which $\mathbf{f}(\mathbf{x})$ dominates $\mathbf{f}(\mathbf{x}^*)$. Thus, the outcome of a Pareto optimization is not one optimal point, but a set of non-dominated, alternative solutions known as the Pareto optimal set [33]. The set of Pareto optimal solutions lies on a curve known as the Pareto optimal front, $\mathcal{P}$ as shown in Figure 3.1(b).

3.2 Optimization Methods

Many methods have been developed for solving multi-objective problems. Most of these methods generate a set of Pareto optimal solutions and employ additional criteria or rules to select one particular solution for the problem. The latter step involves additional information about user preferences and is called the decision making process.

Generally, there are three different types of approaches in solving practical multi-objective problems depending on how the search for the Pareto set and the decision process are combined. These three classes: a priori methods, a posteriori methods, and progressive methods, as their names imply, differ in the stages when user preferences are employed during the optimization process [34, 35]. A survey of multi-objective optimization methods in engineering design can be found in reference 25.

A priori method: Preferences are included prior to the optimization process. The decision maker has to specify his preferences formally by creating a priority ranking of the different objectives involved. Preferences are expressed using an aggregating function which combines individual objective values into a single value. The actual optimization is then conducted on the single measure, ultimately making it a single objective problem. While many a priori methods are available, the weighted-sum approach is the most popular method used and will be further discussed in the next section.

A posteriori method: Preferences are employed the end of the optimization process after the Pareto front has been completely determined. After the candidate solutions have been found, the decision maker then selects a compromised solution from the Pareto optimal set based on his preferences. The main advantage of this method is that the results are independent of any decision making process. The optimization process only needs to be performed once to obtain all the possible solutions in the Pareto set. Compared to other methods, the set of solutions obtained remains the same irrespective of changes in the decision maker's articulation of preferences. Multi-objective evolutionary algorithms (MOEA) fall into this category and will be further discussed in the succeeding sections of this thesis. More information on MOEAs is found in references 34 and 36.

Progressive method: Preferences are used concurrently with the optimization process. During the optimization process, progressive preference information is supplied by the decision maker to guide the search process. This method is a learning process where the decision maker progressively gets a better understanding of problem and interactively refines his preferences to concentrate the optimization effort on promising regions. However, it requires high involvement from the decision maker during the entire optimization process.

3.3 Genetic Algorithm (GA)

3.3.1 GA vs. Conventional Numerical Optimization

There are several advantages that make Genetic Algorithms (GA) superior for solving real-world engineering design problems compared to conventional numerical techniques. GA differs from typical optimization techniques in several ways:

Search from multiple points in parallel: The GA is a global optimizer that works with a population of solutions, searching many directions in parallel and not just a single point. By employing genetic operators, information is exchanged between the multiple peaks and avoids getting trapped in local extremas. Other search methods directionally explore the solution space one point at a time and if the solution found turns out to be suboptimal, the whole process is discarded and re-started again.

Works with both real & discrete variables: GAs can work well on mixed real and discrete variables as well as combinatorial problems because they work with the coding of the parameters and not the parameters themselves. Most conventional search methods are very static and can usually only solve specific types of problems. GAs versatility allows a wider range of problems to be solved.

No need for analytical information: The algorithm only needs the objective function and corresponding fitness levels to guide its search. This makes GAs generally more simple and straightforward to apply because there are no requirements for function derivatives, domain-specific information or other auxiliary knowledge. Calculus-based and gradient search strategies often depend on sensitivity-based information of the objective and constraint functions to guide its search. However, such analytic information is expensive to obtain and often unavailable beforehand.

Robust in complex spaces: Many practical design problems involve nonlinear, discontinuous and multimodal characteristics yet the GA has been shown to be robust in finding an optimal solution in such complex spaces.

Although it does not always deliver a provably global optimal solution to a problem, it can almost always deliver at least a very good solution [37]. On the contrary, conventional search techniques often require a uni-modal, convex and continuous performance landscape of the search space for success.

Works for both single and multi-objective problems: Another area in which GAs excel is their ability to solve not only single objective problems but multi-objective ones as well. In the case of multi-objective optimization problems where there is not just one solution, GAs can effectively identify the set of potential solutions in a single run. The use of parallelism enables GAs to produce multiple equally good solutions to the same problem, possibly with one candidate solution optimizing one parameter and another candidate optimizing a different one [38]. This makes GA effective for solving problems involving conflicting objectives with large search spaces that cannot be accommodated by most traditional search techniques.

Guided by stochastic operators, not deterministic rules: The transitional rules for GAs are of probabilistic nature rather than deterministic. The randomized search is guided by meta-heuristic represented by the fitness value of each chromosome and how it compares to others.

3.3.2 Basic Concepts and Definitions

Evolutionary algorithms are general purpose stochastic search methods inspired by the theory of natural biological evolution. Genetic algorithm is a particularly established class of evolutionary algorithm. It was pioneered in the United States by John H. Holland in the 1970s at the University of Michigan [39]. GA operates on a population of potential solutions by applying Darwinian's principle of survival of the fittest to produce better and better approximations to a solution. The two underlying principles of GAs are selection and variation. Selection mimics the competition for reproduction and resources of individuals whereas variation imitates the natural capability of creating new individuals by means of recombination and mutation. In an optimization problem, selection and variation converts to the exploitation and exploration of the search space. A comprehensive study of genetic algorithms can be referred in Goldberg's book [40].

The basic element processed by a GA is a string formed by concatenating sub-strings, each of which is an encoding of a parameter in the search space also called chromosome. Thus, each chromosome represents a point in the search space. The GA creates a population of solutions and applies genetic operators to selected individuals to evolve the solutions in order to find better ones.

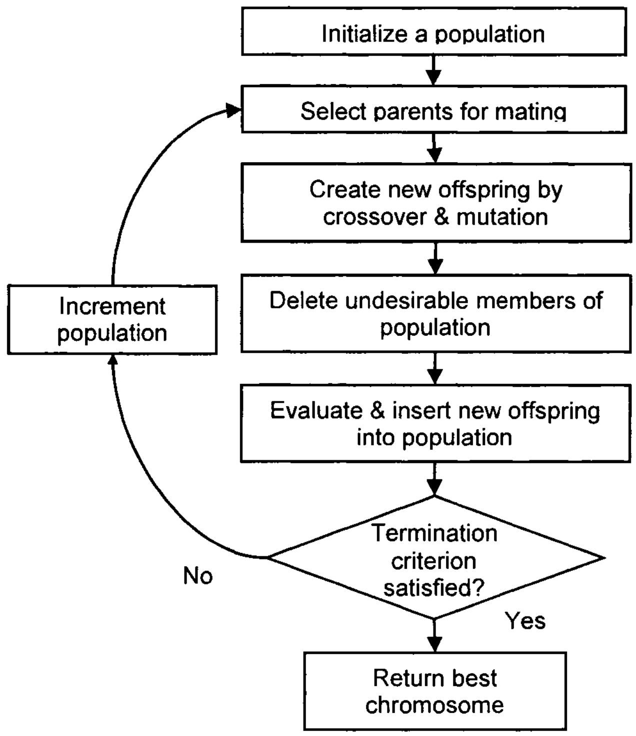


Figure 3-2 General flowchart of the Genetic Algorithm.

The basic steps of a GA are outlined in Figure 3-2. At the beginning, an initial population is randomly initialized. An evaluation function assigns a fitness measure of the individual performance based on the problem to be solved. Individuals are then selected according to their fitness for reproduction. Parents are combined to produce offspring and mutation is carried out on the offspring. The fitness of the offspring is then computed and inserted into the population to produce a new generation, replacing the parents. This cycle continues until a termination criterion is reached. The three most important aspects of using genetic algorithms are definition of the evaluation function, encoding representation of the problem and proper genetic operator settings. A more detailed explanation of the main GA methods and operators is discussed in the following few sections.

3.3.2.1 Selection

In selection, individuals in the population are chosen for reproduction according to their fitness values. Selection determines which individuals are chosen for mating and how many offspring each selected individual produces. The preliminary step to selection is fitness assignment. The fitness function gives a measure of an individual's performance based on the problem to be solved. The fitness function is also alternatively referred to as evaluation function, fitness measure, fitness metric or objective function. Thus, the success of the evolutionary process is heavily dependent on the effective formulation of the fitness measure. Proportionate fitness assignment, rank-based assignment and multi-objective ranking are the typical approaches used for the fitness formulation.

In the actual selection process, each individual in the selection pool receives a reproduction probability depending on their relative fitness in the selection pool. A survey of the general classes of selection methods can be further referred to in references 37 and 41. Among the common techniques are roulette-wheel selection, elitist selection, deterministic selection, tournament selection, rank selection, generational selection, steady-state selection and remainder stochastic sampling. Some of these techniques can be used in conjunction with others. Brief explanation of these techniques is discussed next.

Roulette-wheel selection: The roulette-wheel scheme is a form of fitness-proportionate selection where more fit individuals are more likely, but not certain to be selected. This algorithm is analogous to a roulette wheel whose slots have different sizes that are proportional to the value of the fitness function of every candidate. An example of a roulette wheel with five candidates is illustrated in Figure 3-3. During reproduction, the parent selection process is conducted using the biased roulette wheel where all the candidates in the population are placed. Thus individuals with higher fitness values will be selected more often. Clearly, this method will eliminate the least fit members over the generations and spreads the genetic materials of the fittest population. This scheme however is susceptible to the loss of a best member in the population due to the stochastic nature of the selection process which is called stochastic error.

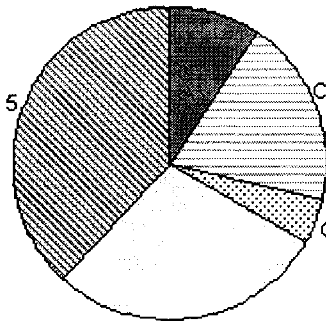


Figure 3-3 Example of a weighted roulette wheel.

Elitist selection: This elitist strategy preserves the best individual by copying the best member of each generation into the succeeding generations. Elitism generally improves the performance of GA because it prevents losses due to stochastic error. However, this strategy may increase the speed of domination of a population by a super individual and thus improves the local search at the expense of global perspectives [38].

Deterministic Selection: In this scheme, the probabilities of selection, p_{select} are calculated proportional to fitness, $p_{\text{select}_i} = f_{\text{fit}_i} / \Sigma f_{\text{fit}_j}$ where $i, j = 1, 2, \ldots, N$ where N is the population size and f_{fit} is the fitness of the individual. Then the expected number e_i of offspring for an individual is calculated by $e_i = N\, p_{\text{select}_i}$. Each individual is allocated offspring according to the integer part of the e_i value. The remaining individuals needed to fill out the population are then drawn from the top of the sorted list.

Remainder Stochastic Sampling: The starting point of this method is identical to deterministic sampling where expected individual counts are calculated as before and integer parts are allocated. In remainder stochastic sampling with replacement, the fractional parts of the expected number values are used to calculate weights in the roulette wheel selection procedure to fill the remaining population slots. In remainder stochastic sampling without replacement, the fractional parts of the expected number values are treated as probabilities instead.

Tournament selection: In tournament selection, subgroups of n_t individuals are chosen randomly from the population with $n_t \leq N$ and compared according to their fitness values. Then the best individual from this group is selected as parent. This process is repeated as often as individuals must be chosen.

Rank selection: Each individual in the population is assigned a rank based on its fitness. The actual selection process is then based on the ranking itself rather than differences in fitness. This method prevents very fit individuals from gaining dominance too early at the expense of less fit ones, which would reduce the population's genetic diversity and hinder attempts of finding the global optimum. However this may also lead to slower convergence since the best individuals do not differ so much from other individuals.

Steady-state and generational selection: In steady-state selection, the offspring of the selected individuals from each generation are placed back into the existing population, replacing some of the less fit members of the previous generation. The number of individuals retained between generations are determined based on the generational gap, G_g allowed where $0<G_g<1$ [42]. In generational selection, when $G_g=1$, no overlapping of the population is allowed. All members of the old population will be deleted and replaced by offspring of the selected individuals.

3.3.2.2 Encoding

Encoding maps a finite-length string to the parameters of an optimization problem and the coding type depends heavily on the problem. The two fundamental guidelines for choosing a GA coding are to create meaningful building blocks and to select minimal alphabets that permit natural expression of the problem [43]. It has been shown that using short and low order building blocks gives high likelihood of success. Large and interdependent building blocks on the other hand may cause GA to fail [40, 44].

Based on the second guideline, binary alphabets offers maximum number of schemata per bit of information compared to any other coding. In binary encoding, every chromosome is a string of bits, giving many possible combinations of chromosomes even with a small string size. Binary encoding is most popular because of its relative simplicity and has been successfully applied in many problems.

In problems that deals with numerous real decision variables, have large extremities or require high precision, binary encoding becomes less effective. The binary chromosome length becomes inherently long, leading to less meaningful building blocks and inhibits the performance of GA. For these problems, direct value encoding is more applicable. In the real value encoding, every chromosome is a sequence of direct representation of real values. This encoding is more challenging to implement as it requires complex crossover and mutation techniques. Therefore, real representations are preferred only when the use of binary encoding is difficult or unnatural. In this thesis, only binary encoding will be discussed.

One popular binary method used for coding multi-parameter problems involving real parameters is uniform coding [44]. For a parameter $x \in [U_{\min}, U_{\max}]$, the decoded unsigned integer is mapped linearly from $[0, 2^l]$ to the specified interval $[U_{\min}, U_{\max}]$. The precision of this mapped coding is given by:

$$\Pi = \frac{U_{\max} - U_{\min}}{2^l - 1}$$

where l is the string length and Π is the precision. In this way, the range and precision of the decision variables can be carefully controlled. Each parameter coding can have individual sub-lengths depending on how much precision is required and are concatenated appropriately to construct the multi-parameter coding.

3.3.2.3 Crossover

Crossover is also referred to as recombination. For consistency, the notion crossover will be used throughout this thesis. Crossover produces new individuals by combining bits and pieces of information contained in the parents from the mating population. Depending on the encoding type of the individuals, different methods for crossover can be applied. During the crossover of binary variables, only parts determined by crossover points are exchanged between the individuals to produce new offspring. The number and types of the crossover points distinguish the methods. Typical methods for binary crossover operations are single-point, multi-point, uniform and shuffle crossover which will be discussed in this thesis. For real valued recombination, the common methods are intermediate recombination, line recombination and extended line recombination.

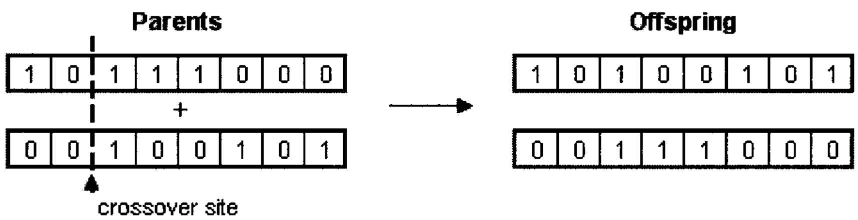


Figure 3-4 Single-point crossover operation.

Single-point crossover: In single-point crossover, one crossover point along the chromosome length is selected at random and the strings are exchanged between the individuals about this point. Then two new offspring are produced. Figure 3-4 illustrates this process. In double-point crossover, two crossover points are selected for exchange of information.

Multi-point crossover: Similarly, for multi-point crossover, n_c crossover points at positions $k_i \leq L-1$ where $i=1:n_c$, and L is the length of the chromosome, are chosen at random for exchange of information. The bits are swapped between the successive crossover points. Figure 3-5 illustrates a three-point crossover process.

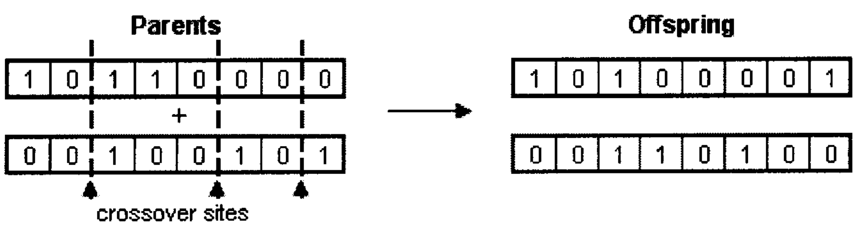


Figure 3-5 Multi-point crossover operation.

This method addresses the idea that important parts contributing most to the performance of a chromosome may not necessarily be in adjacent substrings [45]. The disruptive nature of multi-point crossover also prevents premature convergence of highly fit individuals by encouraging the exploration of the search space [46]. On the contrary, increasing the number of crossover points results in more random mixing and less structure. This degrades the performance of GA as fewer important schemata can be preserved.

Uniform crossover: Uniform crossover is another way to implement multi-point crossover. A crossover template or mask is randomly created to indicate which parent will contribute its parts to the offspring. Figure 3-6 illustrates this process.

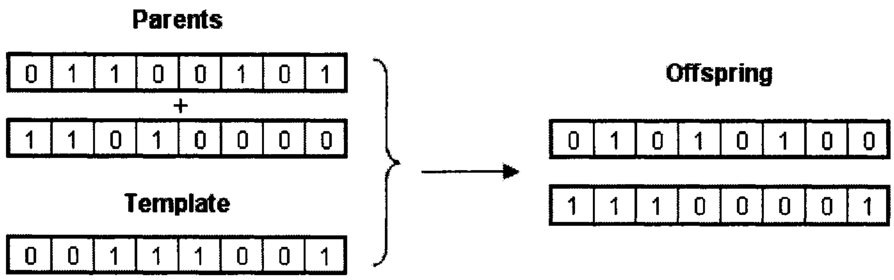


Figure 3-6 Uniform crossover operation.

Re-ordering crossover: Unlike the operators previously discussed, re-ordering operators are used in strings that incorporate both bit values and ordering information. Two points are chosen at random and the chromosome is cut at these two points. At the cut section, the end points switch places. This is also referred to as inversion operator. This type of crossover is mainly employed in permutation encoding where fitness values depend on the string arrangement. Figure 3-7 illustrates this operation.

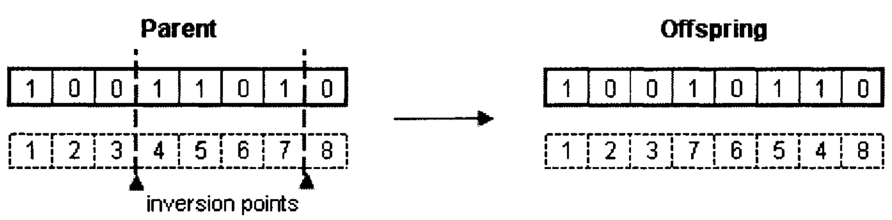


Figure 3-7 Reordering crossover operation.

3.3.2.4 Mutation

Mutation is a random alteration of a string that produces incremental changes in the offspring generated through crossover. By itself, mutation is equivalent to a random search. However, in GA, mutation also helps to prevent premature convergence and prevents the GA from falling into local extremes. The random nature of mutation can introduce new information that were not present in the initial population and can replace the gene values lost from the population during the selection process.

As with crossover, the type of mutation used is dependent on the type of encoding employed and in this thesis, only mutation for binary-coded variables is discussed. For real-coded variables, mutation often involves the random generation of new characters in specified positions. For binary coded variables, the typical method is bit inversion where a bit is randomly selected and inverted as shown in Figure 3-8.

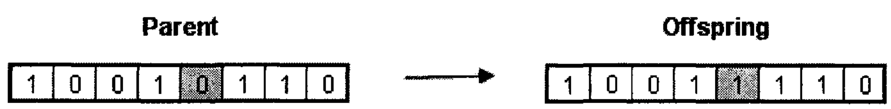


Figure 3-8 Mutation or bit-inversion operation.

3.3.2.5 Parameter Setting

Several studies have been performed to study the effect of the GA control parameters on different problems. Some suggested values given by DeJong [42], Schaeffer [47] and Grefenstette [48] can be found in literature.

The crossover probability, P_c defines how often crossover will be performed. The crossover rate varies for different problems but should be high to encourage mixing. A low crossover frequency decreases the speed of convergence. On the contrary, too high a value may contribute to premature convergence. In general, the recommended P_c should be between 0.6 and 0.95 for increased selection pressure [42, 47, 48]. Another suggestion is to select P_c based on $(n_t - 1)/n_t$ where n_t is the tournament size to avoid disruption [43, 49].

Likewise, the mutation probability, P_m denotes how often parts of chromosome will be mutated. It introduces diversity into the population and should be of a small value to avoid the GA from becoming a random search. DeJong [42] suggested that the mutation rate be inversely proportional to the population size whereas other recommendations are based on the string length [50]. For engineering design problems, some experiments have shown that using the larger value yields the better results [51].

The population size, N defines the number of individuals in the population. A small population size limits the exploration of the search space and inhibits the purpose of crossover operations. Conversely, using a large population size is computationally expensive. Depending on the problem and the type of encoding, many research have shown that after a certain limit it is not useful to use very large populations because it does not solve the problem faster than moderately sized populations. According to the DeJong's [42] standard settings, the suggested population size is 50. Research by Schaffer [47] and Grefenstette [48] has also shown that a population size of about 20-30 is sufficiently good. More recently, research in micro-GA (μGA) has shown that a small population size of 4 to 10 can accelerate fitness convergence and avoid local optima [52-54]. It is typically used when the computational expense of the fitness evaluation favours a smaller population size. Other recommendations are based on population scaling [55] and the size of encoded strings [56]. In general, the proper population size depends on problem complexity and how large the solution space is.

The number of generations, G determines when to terminate the GA operation. The main criterion is to identify when the search has converged, becomes stagnant and has reached a point of diminishing returns. In GA, there is a tradeoff between solution exactness and computational complexity. In terms of computational expense, when the converged population is nearly uniform, terminating the GA avoids wasting resources on the inefficient mutation-based search. One way to determine the near-uniformity of a population is to specify a threshold of minimum change, Δ in the solutions to be considered as converged. In the case of Pareto solutions, the Δ would signify the % change of solutions in the non-dominated set in two consecutive runs [57]. If $\Delta > 0$, the user is accepting less exact representation of the optimal solutions to achieve a decrease in computing time.

3.4 Weighted Sum GA (WS-GA)

The weighted-sum method is the easiest and most popular technique used for solving multi-objective problems. Here, each objective f is assigned a weighting value w and aggregated into a single objective function U:

$$U = \sum_{i=1}^{k} w_i f_i(\mathbf{x}) \quad \text{where} \quad \Sigma_{i=1}^{k} w_i = 1$$

As the objective functions are usually of different magnitudes, the weights often need to be normalized. The relative value of weights generally reflects the relative importance of each objective. Since the weighted-sum method is especially effective when the relative importance of the objectives is known or can be estimated. The decision maker may vary the weights to reflect his preferences before solving the problem. Since the solution is a single Pareto optimal point, systematically varying the combination of weights will also generate the Pareto optimal set.

3.4.1 Handling Constraints

Penalization techniques are the most common way to incorporate constraints into a Weighted-Sum GA (WS-GA) optimization problem. In the penalty method, a constrained problem is converted into an unconstrained one by associating a cost or penalty with all constraint violations. Penalties are used to degrade the fitness rating in relation to the degree of constraint violation and can be defined as follows:

$$f_{fitness}(\mathbf{x}) = U(\mathbf{x}) + R\sum_{j=1}^{m} \Phi[g_j(\mathbf{x})]$$

where Φ is a proper penalty function and R is the penalty coefficient. Using the Powell and Skolnick [58] penalty method, an additional heuristic rule is adopted to distinguish between feasible and unfeasible individuals. Using this method, an unfeasible individual can never have a better value than the worst feasible individual. The WS-GA fitness function becomes:

$$f_{fitness}(\mathbf{x}) = \begin{cases} U(\mathbf{x}) & \text{if } g_j(\mathbf{x}) \geq 0 \quad \forall j = 1,2,..,m \\ U(\mathbf{x}) + R\sum_{j=1}^{m} |g_j(\mathbf{x})|^{\beta} & \text{otherwise} \end{cases}$$

Adjusting the exponent β emphasizes the magnitude of constraint violation so as to considerably exemplify the non-feasibility of a solution.

3.4.2 Disadvantages of Weighted-Sum GA

Although the WS-GA method is simple, there are a several problems associated with its use. A study of some of the drawbacks can be found in [59]. The success of the weighted-sum method is strongly dependent on the proper fitness function formulation and the weight selection. Some of the general problems encountered in engineering design applications are listed as follows:

- *Variability of Preferences*: There is no rigid formula for quantifying qualitative preferences, thus assumptions have to be made for the weight selections. This procedure varies from one person to another, leading to inconsistencies when multiple decision makers are involved.

- *Lack of Information*: When there is a lack of information on the problem to be solved, it is difficult to assign weightings appropriately. In engineering design, there is often little information on the relationships and sensitivities of the objectives prior to solving the optimization problem.

- *Non-Guarantee of Feasible Solution*: A satisfactory a priori selection of weights does not guarantee that an acceptable final solution will be found [59]. If the solution found turns out to be infeasible, the whole optimization process would have to be scrapped and then re-solved with new weights.

- *Improper Penalty Formulation*: Similarly, when constraints are involved, formulation of the penalization function will affect the outcome of the solution. Different use of penalty coefficients may introduce further inconsistencies to the solution. Ineffective formulations may also bias the search to an infeasible solution even if other feasible ones exist.

- *Improper Fitness Scaling*: When multiple objectives are aggregated together, normalization may have to be done in order to handle their different magnitudes.

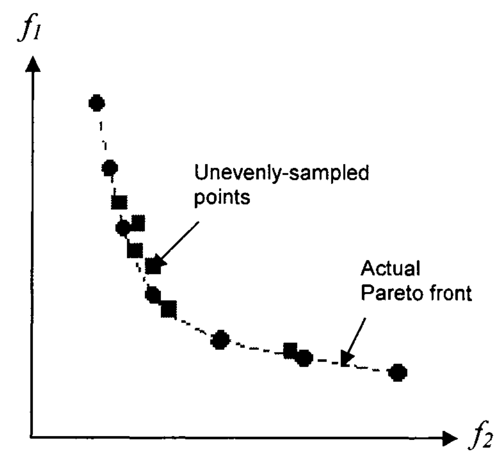


Figure 3-9 Uneven distribution of points on the Pareto front.

- *Uneven Distribution of Pareto Optimal Points*: Another difficulty with this method is that it does not locate the Pareto frontiers consistently and cannot provide an accurate complete representation of the Pareto optimal set [59]. For a given weight combination, only a single point on the Pareto optimal set is obtained. However, varying the weights consistently may not result in an even distribution of Pareto optimal points [32]. This is illustrated in Figure 3-9.

- *No New Knowledge Gained*: At the end of the optimization process, besides just the one solution obtained, no other information is gained. There is no information on how the solution arose or how the objectives interact. Compared to the resources used, the cost of obtaining that one solution is rather expensive especially if the optimization process has to be carried out multiple times.

- *Failure to Find Non-Convex Points*: Finally, it is impossible to locate solutions on non convex portions of the Pareto optimal set in the criterion space using the Weighted-Sum method [59].

3.5 Multi-Objective Evolutionary Algorithm (MOEA)

3.5.1 Introduction

In the conventional WS-GA approach, the optimization converges to one optimal solution. In order to generate the Pareto optimal set, multiple runs have to be carried out by varying the different objective weights. Multi-objective evolutionary algorithms (MOEA) are extensions of GAs that treat multiple objectives separately and try to find a set of solutions that optimizes each objective simultaneously. Since evolutionary algorithms work with a population of solutions, a number of Pareto optimal solutions can be obtained in just a single run. The idea of MOEA is to find as many different Pareto optimal solutions as possible and spread them over the entire Pareto optimal front.

There are many types of MOEAs and their related applications can be found in literature. In MOEAs, fitness assignment is generally based on the concept of dominance ranking [34, 60] or Pareto strength [61, 62] whereas population diversity is usually maintained using fitness sharing or niching [34, 60, 63, 64]. Surveys and comparisons on the different MOEA methods can be referred to in references 41 and 65. The elitist Non-dominated Sorting Genetic Algorithm II (NSGA II) is currently one of the most popular state-of-art method and will be the focus of this thesis.

3.5.2 Advantages of MOEA

In comparison to the WS-GA method, the advantages of using MOEAs to solve multi-objective problems are as follows:

- *Eliminate Inconsistencies in Problem Formulation*: The main advantage is the results are independent of any a priori decision making process. Thus, difficulties associated with user preferences, weights selection and lack of knowledge during the problem formulation is eliminated. Since each objective is treated separately and constraints can be directly handled in MOEA, discrepancies in fitness scaling and penalization methods can also be avoided.

- *Flexibility in Decision Making*: After the Pareto front has been determined at the end of the optimization process, the decision maker can then select a solution from the candidate set based on his preferences. In real world problems, objective priorities often change according to current conditions and this method allows the decision maker to choose a suitable solution accordingly to reflect the changes in preferences.

- *Give Trade-off Information*: MOEAs can provide a set of Pareto optimal solutions that depicts the trade-off between the competing objectives. Having a Pareto front not only allows flexibility in decision making but also gives insight into the system characteristics. Based on the solutions obtained, the decision maker can have a better understanding of the complexity of the problem, the system expectations and the priorities among the objectives before making well-informed decisions or further refining the requirements.

- *Ensure A Good Spread of Solutions*: MOEAs naturally allows niches to persist and can thus preserve the diversity of the Pareto solutions, distributing the solutions evenly across the Pareto frontiers. This way, the early dominance of a particularly fit solution that restricts the scope of the search can be avoided.

- *Capable of Identifying the Pareto Front in One Run*: MOEAs are capable of identifying the possible solutions of the Pareto optimal set in a single run. The optimization process only needs to be performed once compared to the weighted sum method where it has to be iterated using different weights.

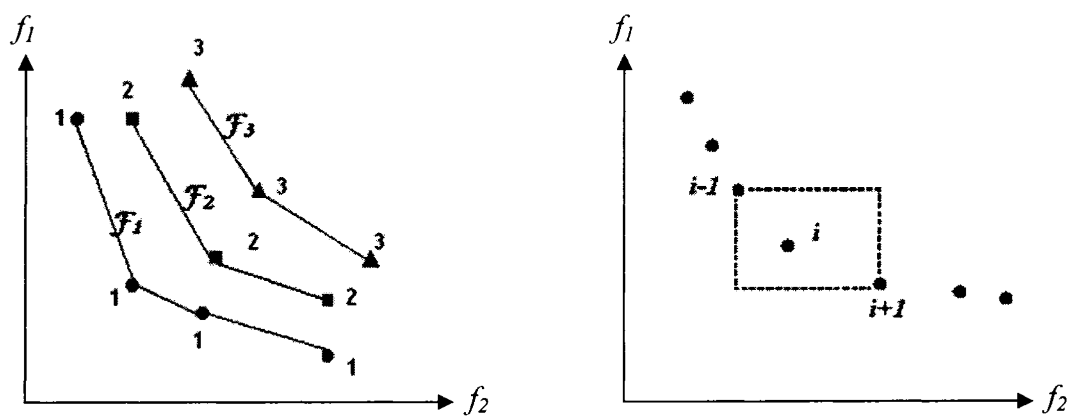
3.5.3 Non-Dominated Sorting Genetic Algorithm II

The elitist Non-Dominated Sorting Genetic Algorithm II (NSGA II) was proposed by K. Deb et al in 2000 [60] and is currently one of the most popular MOEA method used in complex and real-world multi-objective optimization problems. It is an improved version that is considerably different from the original Non-Dominated Sorting Genetic Algorithm NSGA [63].

Some of the distinguishing features of NSGA II are its fast elitist sorting strategy that involves a combined pool of both the parent and child populations and the elimination of sharing parameters using an autonomous crowding distance strategy. NSGA II maintains a Pareto archive and introduces elitism by comparing the current population with the previously found best non-dominated solutions. The selection procedure generally consists of two mechanisms: Non-dominated ranking and crowding distance assignment.

3.5.3.1 Basic Concepts

Non Dominated Ranking: Based on the concept of dominance ranking, each solution is assigned a discrete fitness value equal to its non-domination level with '1' being the best level. These values also indicate the Pareto front, $\mathcal{F}$, to which the solution belongs. Figure 3-10(a) illustrates this concept with the non-domination rank of each point labeled beside it.



(a) (b)

Figure 3-10 (a) The non-dominated ranking procedure and (b) the crowding distance calculation of NSGA II.

Crowding Distance Assignment: The crowding distance is defined as the largest cuboid enclosing the point i without including any other points in the population [60]. Figure 3-10(b) shows the crowding distance of the i^{th} solution as the average side-lengths of the cuboid enclosing it. Since the density of solutions in the neighborhood is represented by the crowding distance, no extra niching parameter needs to be specified. This eliminates the fitness sharing parameter related problems found in several MOEA approaches [34, 63, 64]. This method also preserves boundary points from being lost.

Niched Comparison Operator: The niched comparison operator, $\geq_n$ guides the selection process towards a diverse set of points on the Pareto front [60]. This crowding comparison procedure is performed such that for two solutions of different ranks, the one with a lower rank is preferred whereas for equally ranked solutions, the one with the highest crowding distance value is preferred. This selection criterion between two solutions, i and j can be specified as:

$$i \geq_n j \text{ if } (i_{rank} < j_{rank}) \text{ or if } ((i_{rank} = j_{rank}) \text{ and } (i_{dist} > j_{dist}))$$

where i_{rank} is the non-domination rank and i_{dist} is the local crowding distance.

3.5.3.2 Algorithm

In NSGA II, the initial parent population selection starts out similar to the typical GA but from generation $t \geq 1$ onwards, the procedure differs in the way the selection pool is formed and how new parents are selected for reproduction [60]. The general procedure is described in the following algorithm:

First a combined population $R_t = P_t \cup Q_t$ is formed by combining the parent population P_t and child population Q_t to form a pool size of $2N$. Then the combined population R_t is sorted according to non-domination and the solutions from the first Pareto front is added into new parent population P_{t+1}. If the size of the first front F_1 is smaller than the population size N, then solutions from the next front is added to it and so on until the population size exceeds N. In order to reduce the number back to N, solutions of the last added front are sorted according to the niched comparison operator and then only the first N points are picked for reproduction to create the next generation Q_{t+1}. Here, the crowding comparison procedure is carried out in tournament selections and during the population reduction phase.

3.6 Multi-Objective Optimization for Casting Design

When selecting an optimization algorithm for the casting design problem, several attributes have to be taken into consideration such as reliability, efficiency, generality, and ease of use which are discussed as follows:

- *Reliability*: Since the initial casting design is often unsatisfactory, the algorithm should be able to converge to the optimal point regardless of the initial design(s).
- *Ease of Use*: The algorithm should be easy to use and should not require intensive knowledge and understanding of the mathematical structure of the algorithm. Since knowledge about the search space in the casting design problem is often unavailable beforehand, the algorithm should not require extensive specification and/or tuning of parameters.
- *Generality*: The algorithm must be general and should not impose any restrictions on the form of the functions or the constraints in the different casting problems. It should be generally applicable to various casting design cases.
- *Efficiency*: An efficient algorithm has a faster rate of convergence to the optimum point(s) and requires less number of design evaluations. Since casting design is a multi-objective problem, it is preferable to obtain a set of Pareto optimal solutions with the least number of design evaluations to facilitate flexibility in decision making.

Since the GA is a global optimizer that conducts its search from multiple directions, it has the ability to find an optimal solution even if the choice of starting point is unfeasible. Using GA can avoid the difficulty of choosing a suitable initial design by trial and error which is necessary in gradient-based methods. Furthermore, the casting design problem is characterized by complex relationships between the objective functions and the gating and riser components (e.g. shapes, sizes, locations and quantities). Thus, the stochastic nature of GA makes it suitable for this application.

The key aspect of using GA for the casting design problem is that it does not require any analytical information for the optimization formulation. In the casting design problem, sensitivity-based information is both difficult and expensive to obtain due to the time consuming design evaluations. Using the GA technique, there is no need for function derivatives since the algorithm only uses the objective function and fitness levels to guide its search. This makes the GA generally more straightforward to apply to different casting design problems without the need for extensive customization of the algorithm each time.

Another motivation for using GA to solve the casting problem is its versatility for solving both single and multi-objective problems. In GA, the advantages of using a scalar or vector optimization approach for the casting design problem can be explored using the WS-GA and MOEA methods. Finding the proper weights and constraint formulations in WS-GA can be difficult without any prior knowledge of the performance landscape of the search space. Therefore, the MOEA approach is also chosen for this study to address that difficulty. Inconsistencies in the problem formulation can be avoided because MOEA treats each objective separately and handles constraints directly. Furthermore, using the crowding distance factor in NSGA II can eliminate the need for extra niching parameters which can introduce further inconsistencies.

Finally, the MOEA approach has the ability to obtain multiple solutions in one run, compared to just a single point in other methods. Since finding a compromise between quality and cost is an important decision factor in the casting design problem, using MOEA can give the advantage of generating a tradeoff curve between the competing objectives.

In this study, both the WS-GA and MOEA (using NSGA II) methods were implemented to compare their feasibility for the gating and riser system design. Their efficiency is also compared with the popular Design-of-Experiment (DOE) method.

Chapter 4

Optimization of Gating and Riser Design Using Evolutionary Algorithms

Evolutionary algorithms were employed in the optimization framework to demonstrate their capability and robustness in the complex casting problem. In this chapter, the problem formulation of the gating and riser system design is described. An approach for integrating the optimization framework with a commercial simulation software for design analysis is presented.

4.1 Optimization Problem Formulation

Many aspects have to be taken into account when formulating the casting design problem as an optimization problem as it involves translating the problem into a well defined mathematical statement. *'What are the goals?'*, *'What are the design variables?'*, *'How to select the best design?'* and *'What are the terminating criteria?'* are questions that need to be addressed [32, 66]. Some general guidelines for the formulation of practical design optimization problems can be found in reference [32]. In general, formulating the casting design optimization problem can be summarized into four steps: Firstly, the objectives and requirements of the design problem have to be identified. Next, free variables that can be manipulated during the optimization process to produce different designs are identified as design variables. Then a comparison measure to determine how good a design is must be established. Finally, a terminating criterion to end the optimization process is specified. Here, generalized formulations of the design variables, objectives and terminating criterion are described.

***Design Variables*:** In the gating and riser design optimization process, different designs are explored by changing the shapes and sizes of the gating and riser components. Thus, geometrical descriptors of those components such as length, radius, and height are the commonly used design variables. When dealing with geometrical shapes, dimensional constraints are also usually imposed to the design variables for maintaining realistic shapes.

***Design Objectives*:** In the design of a gating and riser system, two main objectives should be taken into account: (1) eliminating casting defects and (2) maximizing casting yield. Because a commercial software, MAGMASOFT® is used for the gating and riser system design analysis, there are no details on the complex calculations involved. Therefore, some of these objectives have to be measured directly from the accessible simulation outputs. In this study, the quality of a cast product can be characterized by its shrinkage porosity and liquid metal velocity.

Using the output results available from the MAGMASOFT® simulation, the shrinkage porosity measure is taken as the maximum porosity value contained in the control volumes of the casting. Since a shrinkage porosity-free casting is desired, the porosity requirement is $P = 0\%$.

Turbulence of the liquid metal in the casting is assessed in terms of flow velocities obtained also from the simulation results. Based on Campbell's rules [31], the liquid metal flow should not exceed a velocity of 0.5 m/s to maintain the stability of the meniscus front. Since the critical velocity V_{crit} was not properly defined in either the X, Y or Z directions [31], the constraint on the entry velocity of the liquid metal is set such that it must not exceed 0.5 m/s in any direction of the velocity vector, V_x, V_y and V_z for a design to be considered feasible. For the purpose of scoring, the magnitude of the velocity vector V_{xyz} is calculated based on the 3-dimensional velocity components, V_x, V_y, and V_z in each control volume of the casting with $V_{xyz} = \sqrt{V_x^2 + V_y^2 + V_z^2}$. The maximum velocity magnitude gives the velocity objective measure.

The metal yield can be calculated based on the volume ratio of the actual casting, Vol_{cast} over the total gating and riser system, $Vol_{gating+riser}$:

$$Yield\% = \frac{Vol_{cast}}{Vol_{cast} + Vol_{gating+riser}} \times 100\%$$

Since minimization of the objective function is chosen, the yield loss, $Y^l = 1\text{-}Y$ was used for the objective measure. Another area of interest in some problems but of less importance is the ease of removing the gating and riser components. The size of the riser ang gating connections must be small compared to the connected portions of the casting to avoid breakage or cracks in the casting during fettling. These connecting parts have to be broken off or machined away after the casting has solidified, therefore complicated connections should be avoided to reduce the removal costs. For the ease of removal objective R, this can be measured as the dimensions of the intersecting area between the component and the casting.

Terminating Criterion: The number of generations, G is used as the termination criterion so that time constraints can be included in the optimization process. By using G, there is a tradeoff between finding the true global optimum and saving time by accepting near-optimal solutions which is usually the case in practical applications. However for the casting design application, the importance of finding the true global optimum is not that critical.

4.2 Proposed Optimization Framework

In this optimization framework, MAGMASOFT® was employed for the simulation work and analysis of the casting gating and riser system design. Here the optimization strategy has to be connected to the simulation environment. The design analysis software is treated like a 'black box' with an input interface to accept new design suggestions and an output interface to communicate their performance measures to the optimization engine. These output measures are used to evaluate the fitness of a design in WS-GA or MOEA. Based on the evaluation results, the optimization algorithm will then determine its next direction and the cycle continues until a termination criterion is met. The optimization framework and its process flow are illustrated in Figure 4-1.

The coupling of the optimization algorithm and the simulation software is done by programming in ANSI-C and mainly involved the execution of system commands, reading and writing to external data files as well as passing arguments between subroutines. The structure of the casting design optimization implementation is divided into four main parts: pre-processing, simulation, post-processing and optimization which are described in the following sections.

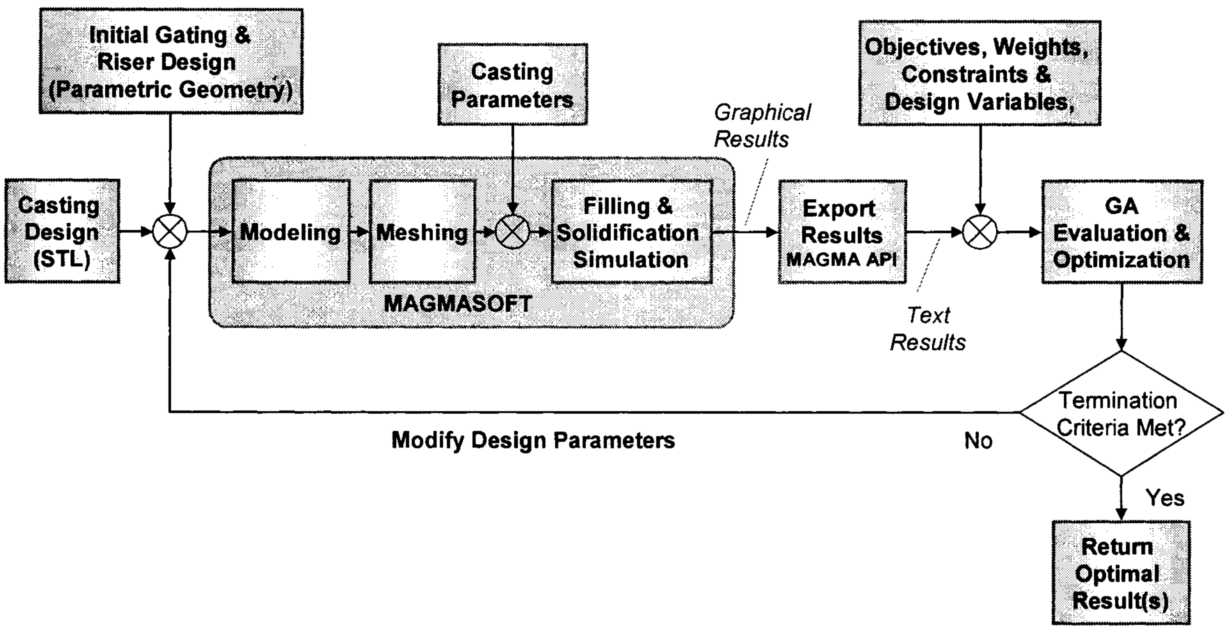


Figure 4-1 Optimization process flowchart of the gating and riser system design.

4.2.1 Pre-Processing

Modeling: Simulation of the cavity filling and solidification process requires the geometrical information for the casting, the gating system and the mould in advance. Firstly, solid CAD models of the casting can be created using any CAD software and converted into STL files. The preprocessor module of MAGMASOFT® then reads the STL files as geometry inputs into the software. The casting shape can also be constructed using modeling functions within the MAGMASOFT® environment.

After the casting model is established, the initial design of a gating and riser system is created using parametric geometry functions. This structured modeling of geometry is done with the aid of command files. By modifying the parameter values in the geometric functions, the design can be varied accordingly. This is an essential part of the strategy to explore different gating and riser designs in an autonomous fashion.

***Meshing*:** Since MAGMASOFT® is a numerical simulation tool that employs finite-difference calculations, the modeled geometry of the full casting system needs to be further divided into individual control volumes prior to the simulation. This subdividing of the geometry into meshed elements is described as the meshing process and can be performed using the enmeshment module in MAGMASOFT®. Depending on the complexity of the model and resolution of the mesh generation, the number of meshed elements can be adjusted accordingly for the desired accuracy.

4.2.2 Simulation

***Casting Parameter Specification*:** Before the simulation can be run, casting process information must be defined. This includes the thermophysical properties of the cast and mould materials and their initial temperature conditions. The heat transfer coefficients also have to be defined for the boundary condition of the materials. Specifications for the filling and solidification process include the filling time or pouring rate, the filling direction, the feeding effectivity, criterion temperatures and the solver types. The feeding effectivity defines the maximum ratio of the volume available for feeding and the actual volume of the riser. The filling time varies from one problem to the other, depending on the casting size. The fill direction indicates the flow of metal into the mold and is defined here in the negative Z direction to match the orientation of the gating and riser system. The filling and solidification simulation parameters used in this study are listed in Tables 4-1 and 4-2.

***Filling and Solidification Simulation*:** Once the meshed geometries and the necessary process parameters have been established, the actual filling and solidification simulation can be carried out. The type of numerical calculations employed is based on the algorithm (Solver) type chosen. Solver 3 is used for speed and accuracy. For both the filling and solidification simulation, results were extracted in 10% increments of the process completion for further analysis.

Table 4-1 The filling simulation parameters

Filling Definition	
Parameter	Value
Solver	Solver 3
Filling Depends on	Time
Filling Time	12 s
Storing Data	10% increments of % Filled
Fill Direction-X	0
Fill Direction-Y	0
Fill Direction-Z	-1

Table 4-2 The solidification simulation parameters

Solidification Definition	
Parameter	Value
Temperature from Filling	Yes
Solver	Solver 3
Stop Simulation	Automatic
Stop Value	425°C
Calculate Feeding	Yes
Feeding Effectivity	70%
Criterion Temperature	442.6°C
Criterion Temperature	603.0°C
Storing Data	10% increments of % Solidified

4.2.3 Post-Processing

Export Results: With the 3-D post processor module in MAGMASOFT®, the visualization of the fluid flow and temperature field patterns in the cavity during the casting process can be graphically analyzed. However, in order to formalize the casting design optimization process, these results have to be converted to a proper format that can be routinely handled. Here, the Application Programming Interface (API) of MAGMASOFT® is employed to export the graphical results into text results. Using the API, customized subroutines were developed to access the MAGMASOFT® files and data structures, extract the desired results from each control volume and convert them into the appropriate format for further processing.

***Result Processing*:** Since the exported results contain vast amounts of values corresponding to the large number of control volumes used in the simulation, additional computations are performed to analyze the results. Then the processed results are further organized to form the appropriate performance measures required for the optimization performance evaluation process.

4.2.4 Optimization

***Problem Definition and Requirements*:** In the problem definition, all the necessary information pertaining to both the optimization algorithm and the casting design problem are defined here. Optimization-related information includes first and foremost, the objective functions, their weights if applicable and the constraints. Next, the algorithm parameters that need to be defined include the WS-GA and MOEA operators, such as crossover and mutation rates, population size, termination criteria etc. Context-dependent casting design parameters include the design variables, their geometrical limitations such as upper and lower bounds and shape constraints if applicable. These information define the overall optimization problem and is imperative for the success of the casting design optimization process.

***Performance Evaluation*:** Here, the performance evaluation of a design is carried out against the objective functions and constraints specified in the problem definition. The fitness is determined based on the type of algorithm used and the output measures acquired from the MAGMASOFT® simulation results. The evaluation process determines the next direction of the optimization process or whether the termination criterion is met.

4.3 Gating and Riser Design Case Study

Test Casting: A cylindrical housing model was used as the test casting to demonstrate the different optimization strategies. The three-dimensional CAD model of the test casting is shown in Figure 4-2. It has an outer radius of 260 mm and 160 mm at the largest and narrowest part, an inner radius of 120 mm and 180 mm at the upper half and bottom part respectively and a height of 245 mm. This casting is relatively large with a total weight of 30 kg.

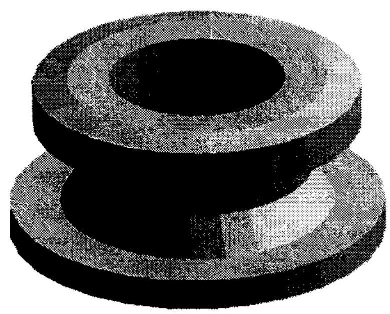


Figure 4-2 3-D model of a cylindrical housing casting.

For this casting, bottom filling of the mold was employed. A tapered sprue was used and metal was introduced into the casting cavity using two ingates. Two equal risers were added to the top of the housing model and two additional risers were added at the circumference near the bottom part as shown in Figure 4-3.

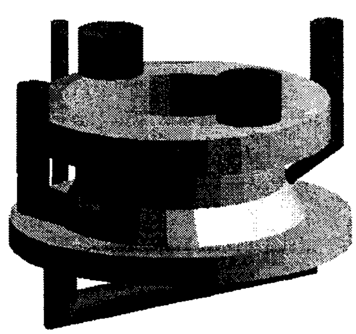


Figure 4-3 The gating and riser components of the test casting.

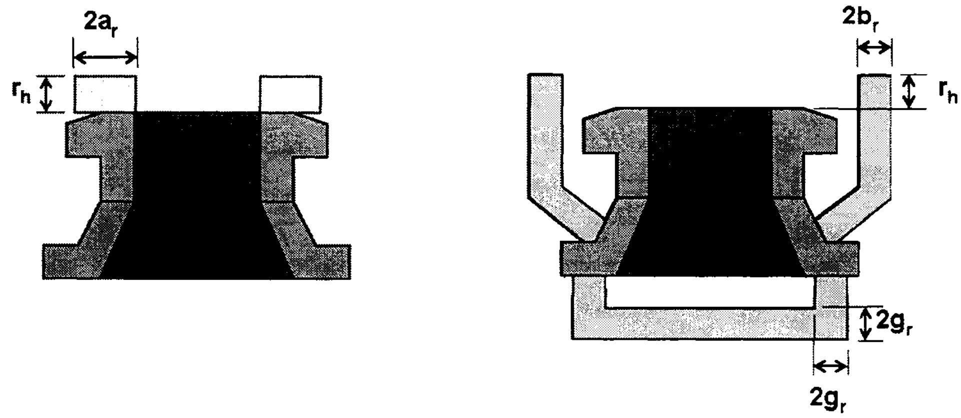


Figure 4-4 Design variables of the housing casting.

For this experiment, cylindrical shaped gating and riser components were used for simplicity. With cylindrical shapes, modifying its cross section area only requires changing the radius parameter. Due to the symmetry of the test casting, the number of design variables can be reduced to four independent parameters. The design variables are the radii of the top risers a_r, middle risers b_r and their heights r_h as well as the radii of the ingates and runner g_r as shown in Figure 4-4. Dimensional constraints were also imposed to the design variables to maintain realistic shapes. The parameter ranges of the design variables are given in the Table 4-3.

Table 4-3 Design variables and the parameter range

Parameters (mm)	a_r	b_r	g_r	r_h
Upper Bound	40	10	12	20
Lower Bound	70	40	35	100

The radius of the side risers and ingates are used as the measure for the ease of removal objective, R. The design requirements are P = 0%, Y > 70% and V_x, V_y and V_z < 0.5 m/s for a design to be considered feasible. The optimization problem for this study can be represented as follows:

$$\text{Minimize } \mathbf{f}(\mathbf{x}) = \begin{bmatrix} V_{xyz}(\mathbf{x}) \\ P(\mathbf{x}) \\ Y^{l}(\mathbf{x}) \\ R(\mathbf{x}) \end{bmatrix} \quad \text{subject to } \mathbf{g}(\mathbf{x}) = \begin{bmatrix} V_x, V_y, V_z(\mathbf{x}) < 50\,\text{cm/s} \\ P(\mathbf{x}) = 0\,\% \\ Y^{l}(\mathbf{x}) < 30\,\% \end{bmatrix}$$

$$\text{where } \mathbf{x} = \begin{bmatrix} a_r & b_r & g_r & r_h \end{bmatrix}^T$$

***Computer Simulation*:** The gating and riser system as well as the housing model were created using the preprocessor module of MAGMASOFT®. The gating and riser system was constructed using parametric geometry functions that could be programmatically modified using command files. The enmeshment of the geometry was made up of approximately 110,000 elements. The resolution of the meshed model could be refined using a larger number of elements. However, this would lead to an exponential increase in the time required for design analysis. Therefore, the number of elements chosen for this study was of an amount that could sufficiently maintain the mesh quality of the model without causing any error or warning messages during the enmeshment procedure.

Magnesium alloy AZ91 was used as the casting material and dry silica was used for the sand mould. The thermophysical properties of both the cast materials were available in the database module of MAGMASOFT®. The initial and boundary conditions used for the simulation are listed in Tables 4-4 and 4-5 respectively. The filling time was 12 seconds. The simulation was carried out on the SunSoft Solaris 8 UltraSPARC-II platform (500MHz, 256MB RAM, 15GB HD) and the time required for each design evaluation, T_{eval} was approximately 13 minutes. Compared to T_{eval}, the CPU time used for the evolutionary algorithm operators were negligible.

Table 4-4 Initial conditions used for computer simulation

Material Definition		
Material Group	Material	Initial Temperature (°C)
Cast Alloy	AZ91	650
Sand Mould	DRY SILICA	20

Table 4-5 Boundary conditions used for computer simulation

Boundary Conditions	
Material Group Pairs	Heat Transfer Coefficient (W/m^2K)
Cast Alloy - Sand Mould	800
Gating – Sand Mould	800
Feeder - Sand Mould	800
Inlet – Sand Mould	800

***EA Implementation*:** Since the accuracy of the model is limited by the meshed elements in the design analysis, a high degree of accuracy in the design variables was not required. Therefore, a string size of 20 was used for the binary encoding. In the implementation of both WS-GA and MOEA, the mutation rate was set at 0.05, which was inversely proportional to the string length to introduce diversity into the population without causing too much disruption [50]. Based on some recommendations found in literature, the crossover rate should be between 0.6 and 0.95 for increased selection pressure [42, 47, 48]. A high crossover rate is preferred to encourage mixing. However, too high a value would lead to premature convergence. Therefore, a crossover rate of 0.8 was chosen for this study. Single point binary crossover and bit-wise flip mutation operators were used. The binary tournament size used was 2 for selection.

The initial population for both the WS-GA and MOEA implementation was randomly generated with a seed of 0.4. In WS-GA, constraints were incorporated using the Powell and Skolnick [69] penalty method with the exponent β=2 and a penalty coefficient of 100 to discourage non-feasible results. The number of generations, G was used as the terminating criterion where it is set according to the maximum number of designs allowed by the total acceptable run time.

Table 4-6 Objective weight setting

Objective Weights	Yield (Y)	Porosity (P)	Velocity (V_{xyz})	Ease of Removal (R)
Set #1	0.15	0.50	0.30	0.05
Set #2	0.20	0.40	0.40	0.0
Set #3	0.25	0.35	0.40	0.0
Set #4	0.15	0.60	0.25	0.0
Set #5	0.33	0.33	0.33	0.0

Population sizing for this study was chosen based on the discussion presented in Section 3.3.2.5. According to the DeJong's [42] standard settings, the suggested population size is 50. Latter research by Schaffer [47] and Grefenstette [48] has shown that using a moderate size of about 20-30 could solve the problem faster without excessive computational costs. A smaller population size of 4 to 10 was also recommended in some recent research on micro-GA to accelerate fitness convergence [52-54].

Therefore, three different population sizes of 10, 20 and 50 were used to observe the convergence and effectiveness of finding the optimal solutions using WS-GA. The WS-GA was also tested using different weight settings shown in Table 4-6 to compare the solutions found. For MOEA, NSGA II was used with three different population sizes of 12, 20 and 40 to compare their ability to find the Pareto front. The optimization results of the WS-GA and MOEA methods are presented in the next chapter.

Chapter 5

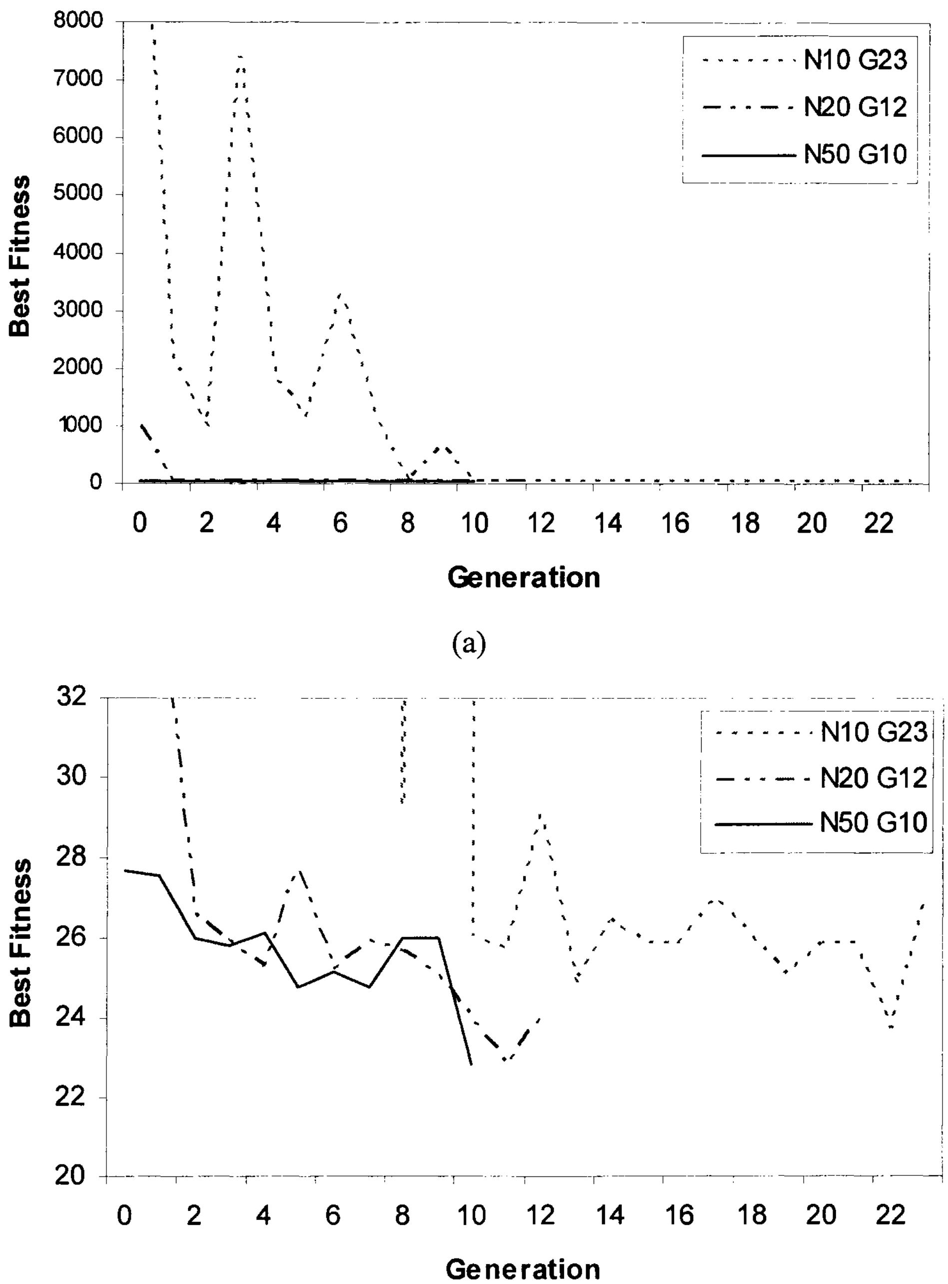
Results and Discussions

Using the proposed optimization framework, two different optimization strategies: WS-GA and MOEA (using NSGA II) were applied to the gating and riser system design of the test casting. The effects of population sizing and weight settings on the optimization performance were studied. The feasibility of using evolutionary algorithms was also compared with a popular design-of-experiment (DOE) method. Discussions of the optimization results using the different approaches are presented in this chapter.

5.1 Optimization with WS-GA

5.1.1 Effects of Population Sizing

To examine the effects of population size on the performance of the algorithm, WS-GA was carried out with three different population sizes of 10, 20 and 50 based on the pre-defined weights in Set #1. The constraints formulated into the objective function were P = 0%, V_x, V_y and V_z < 50cm/s and Y > 70%. For each population size, the best fitness value found in each generation is shown in Figure 5-1. With population size of N=10, the best fitness found in the early generations were inferior due to its small pool size but improved in the subsequent generations when the pool of good building blocks became more established. Doubling the population size to N=20, the best fitness value improved more than with N=10. Increasing the population size to N=50 allowed a comparable good solution to be found in the first generation itself since it had a larger initial pool size available. Eventually, all three population size were able to find comparable solutions in succeeding generations but the best optimal result was obtained with N=20. The optimization results are as shown in Table 5-1.



(a)

(b)

Figure 5-1 Best fitness values in each generation of the WS-GA population: (a) full scale and (b) zoomed-in scale.

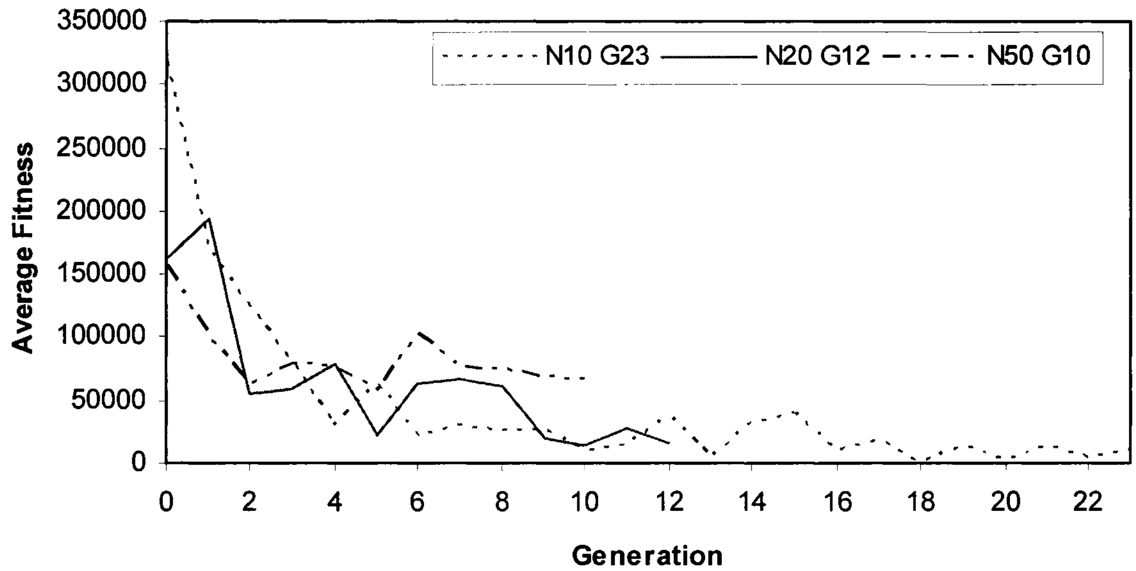


Figure 5-2 Average fitness values in each generation of the WS-GA population.

Figure 5-2 shows the average fitness values of the population in each generation where a population made up of better solutions led to a decrease in the average fitness. For the same reason mentioned earlier, N=50 had an overall higher average after G=5 compared to N=10 and 20 because with the larger population size, it had more combinations of solutions that were still undergoing exploration and not yet dominated by good solutions. From Figure 5-2, it can also be seen that N=10 had the better average fitness convergence compared to N=20 as the generation number increased. Because of its small population size, the whole population could be easily dominated by the fitter solutions. However, this also led to the limited exploration of other designs and resulted with a sub-optimal result.

5.1.2 Effects of Weight Setting

A change in the objective weight setting in WS-GA reflects a change in user preferences or objective priorities. In the second set of experiments, the WS-GA was carried out with different weight settings given in Table 5-4. The constraints formulated into the objective function were P = 0%, V_x, V_y and V_z < 50cm/s and a tightened yield requirement Y>85%. When using WS-GA, changes in the design requirements have to be formulated into a new objective function and the optimization has to be re-run again. The optimization results obtained are shown in Table 5-2.

From the results, it can be seen that some of the optimal solutions obtained at the end of the WS-GA runs were infeasible. In weights Set #2 and #3, although a higher preference was given to the velocity objective, the optimization resulted in a violation of the velocity constraint. On the contrary, an equal weight setting in Set #5 resulted in a violation of the yield constraint.

There is no clear definition of what a good weight setting should be since there is no information on how the objectives interact or how the solution arose. Assumptions had to be made for the weight selections since there is no rigid formula for quantifying qualitative preferences. After trial and error, a weight setting of $V_{xyz} = 0.25$, $Y = 0.15$ and $P = 0.60$ as given in Set #4 managed to obtain a feasible result.

It is obvious that even when weights are assigned to specify the relative importance of each objective, the optimization results may not necessarily reflect the desired outcome. This is because the opposing velocity, yield, and porosity objectives have different magnitudes and characteristics, thus comparing them is extremely difficult if not impossible. With conflicting objectives, the final result is dependent on the relative objective weights and constraint formulation. Since the search is guided by the single objective function, the final solution could be infeasible even when a feasible solution exist. This is especially the case when the degree of constraint violation is relatively small and/or when the pool size is too small for sufficient exploration to take place in the correct direction. If the solution found turns out to be infeasible, the optimization process would have to be re-solved with new weights.

The effects of objective weight settings can be seen in Figure 5-3. In Set #1, with a relaxed yield constraint of 70%, the convergence of the porosity and velocity objectives were faster whereas the yield objective showed a decreasing trend. This is associated with the higher priorities given for the porosity and velocity objectives. With a heavier weight assigned to the yield objective given in Set #2, the yield objective pattern shown in Figure 5-3 demonstrated an increasing trend instead.

Table 5-1 Optimization results using WS-GA with different population sizes

Experiment (WS-GA)	Best Design with Relaxed Constraints (V_x,V_y,V_z <50cm/s, P=0% & Y>70%)												Gen. #*	# of Evaluations*
	Variables (mm)				Objective							Weighted Score (Set #1)		
	a_r	b_r	g_r	r_h	V_x (cm/s)	V_y (cm/s)	V_z (cm/s)	Velocity (cm/s)	Porosity (%)	Yield (%)	Fettling			
N=10 G=23	54.2	11.9	34.4	61.3	42.2	32.6	28.8	60.6	0.0	78.2	46.3	32.23	22	220
N=20 G=12	57.1	11.0	32.8	61.3	39.1	37.5	26.2	58.6	0.0	79.3	43.7	31.66	11	220
N=50 G=10	50.6	12.9	32.0	48.4	39.1	39.5	20.0	59.1	0.0	80.7	44.9	32.07	10	500

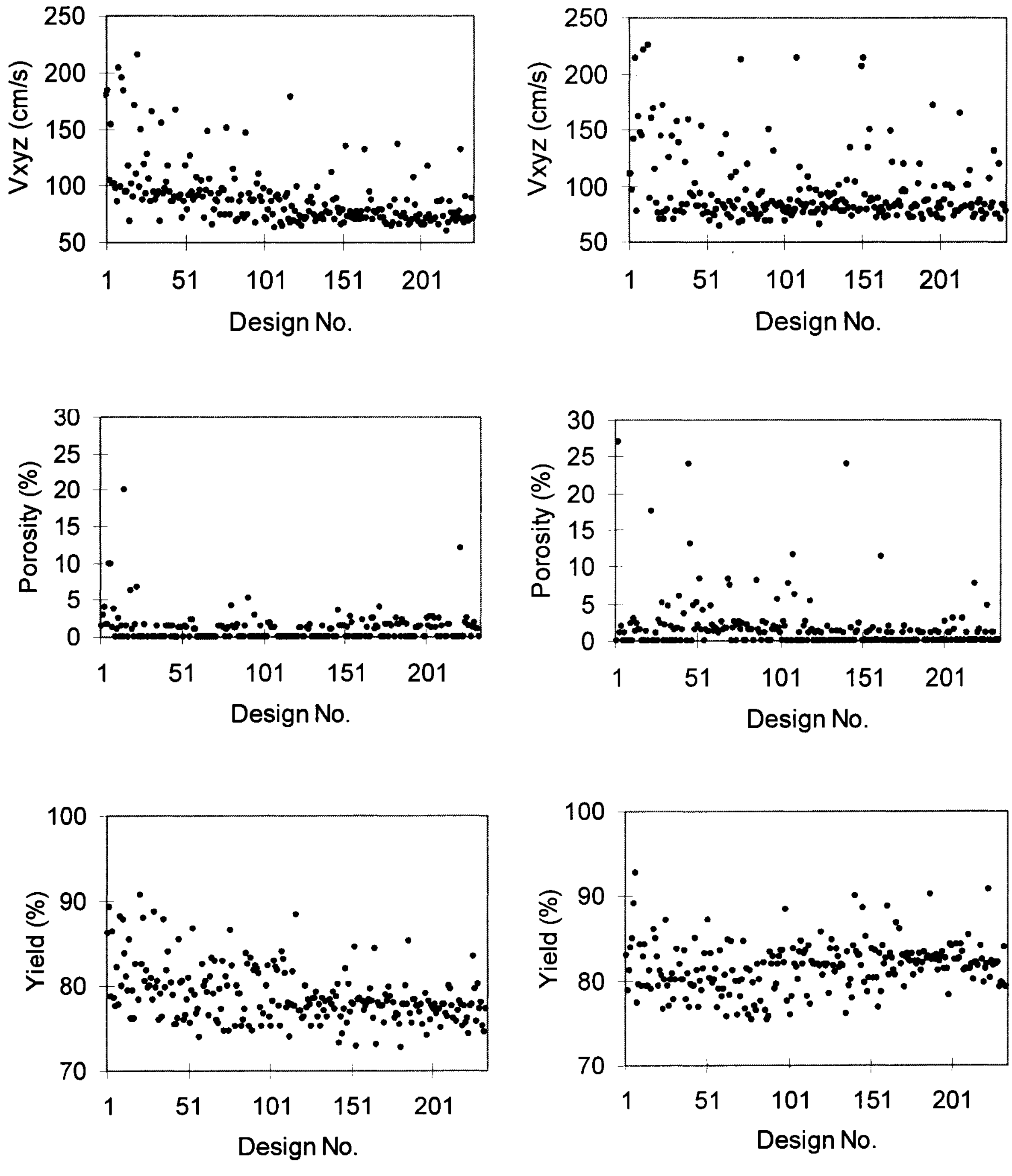
N = Population Size , G = Generation number

**when best solution was found*

Table 5-2 Optimization results using WS-GA with different weights settings

Experiment (WS-GA) N=20 G=12	Best Design with Tight Constraints (V_x,V_y,V_z<50cm/s, P=0% & Y>85%)															
	Variables				Objective Weights				Objective Measure							Feasible?
	a_r	b_r	g_r	r_h	Yield	Porosity	Velocity	Fettling	V_x (cm/s)	V_y (cm/s)	V_z (cm/s)	Velocity (cm/s)	Porosity (%)	Yield (%)	Fettling	
Set #2	54.5	14.8	24.6	61.3	0.20	0.40	0.40	0.0	46.0	46.5	60.1	*88.8 (10271.1)**	0	85.1	39.4	No*
Set #3	54.5	18.7	28.3	56.1	0.25	0.35	0.40	0.0	39.6	51.7	32.2	*72.65 (350.7)**	0	85.2	47.0	No*
Set #4	61.0	11.0	25.4	38.1	0.15	0.60	0.25	0.0	44.0	42.9	39.0	72.7	0	85.7	36.4	Yes
Set #5	66.1	20.6	26.8	48.4	0.33	0.33	0.33	0.0	46.2	44.5	48.7	80.5	0	*82.3 (746.7)**	47.4	No*

**Objective constraint violated are marked by italics & the penalty score is shown in brackets*



(a) (b)

Figure 5-3 Velocity, porosity and yield results using WS-GA (N=10) with different weights: (a) Set #1 and (b) Set #2.

5.2 Optimization with MOEA

Using the MOEA technique, the velocity and yield objectives were considered for optimization with constraints V_x, V_y, V_z < 50cm/s and Y > 70%. The porosity objective P=0% was also treated as a constraint that must be satisfied for a solution to be feasible.

Using NSGA II with N=20, a set of non-dominated solutions was found in each generation. Since constraints are explicitly handled, the solutions found using NSGA II are ensured to satisfy all the design requirements. The shift in the Pareto front found in G = 1, 4, 8, 12 and 16 is illustrated in Figure 5-4. It can be seen that the solutions improved as the generations progressed while better solutions were being discovered just like in WS-GA. At the completion of one run, several Pareto optimal solutions were identified. The solutions found had a good spread that gave trade-off information between the competing velocity and yield objectives as shown in Figure 5-4.

5.2.1 Effects of Population Sizing

NSGA II was used with three different population sizes of 12, 20 and 40 to compare their ability to find the Pareto front. The comparison of different population sizes is shown in Figure 5-5. Better solutions were obtained using N=20 compared to G=40 for the same reasons as in WS-GA because more evolutionary operations were allowed to take place in N=20 with the same amount of design evaluations. With N=12, the solutions found were also dominated by the solutions found with N=20. With N=12, the optimization converged quickly but the limited pool size restricted the GA crossover exploration and impeded the discovery of new and better solutions.

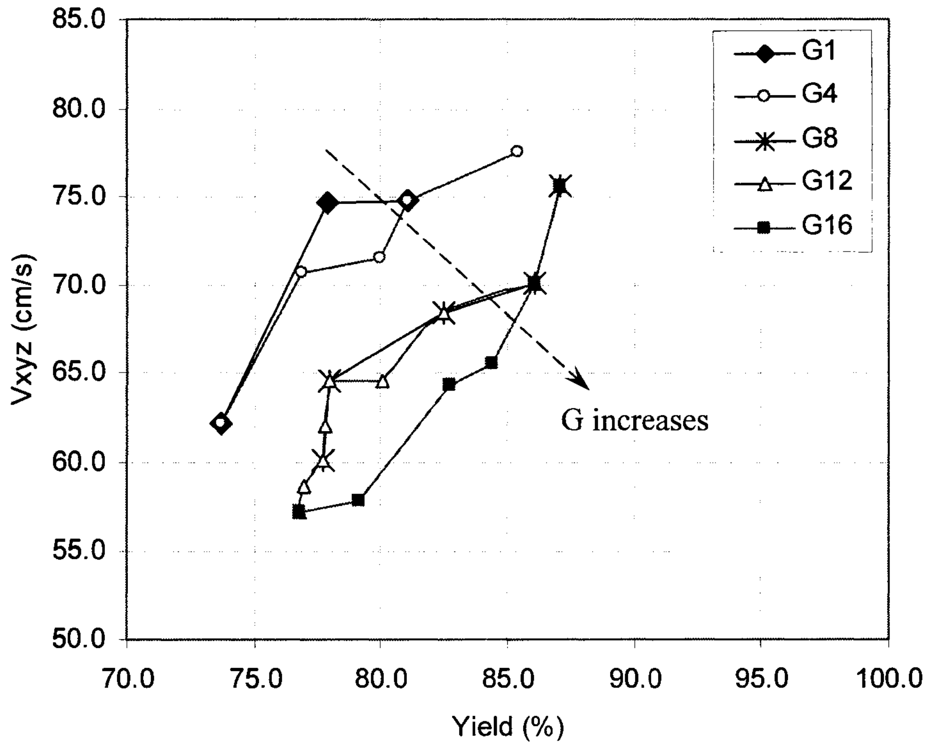


Figure 5-4 Shift of the Pareto front in different generations (NSGA II, N=20).

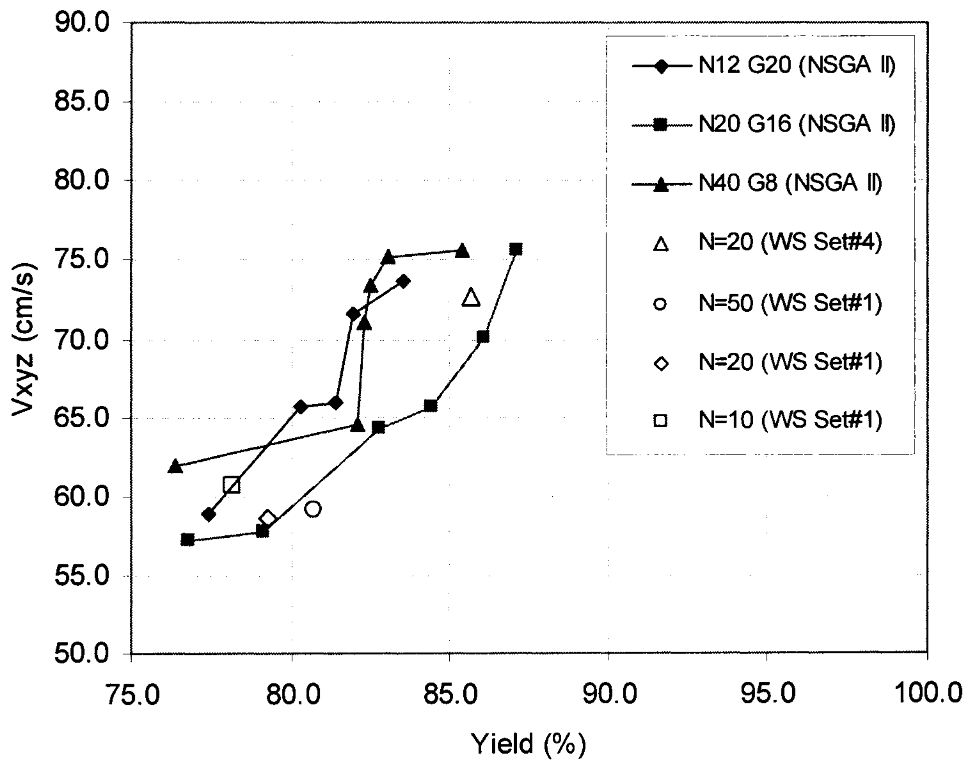


Figure 5-5 Pareto solutions found in NSGA II compared to WS-GA.

5.2.2 Effects of Requirement Change

Using MOEA, changes in the design requirements did not require any re-runs or modifications to the optimization algorithm which was necessary when using WS-GA. This is because requirements and preferences were not involved in the fitness formulation and thus did not affect the search results.

With the design requirements constraints of P = 0%, V_x, V_y and V_z < 50cm/s and a tightened yield requirement Y > 70%, a solution can be picked from the set of solutions found in Figure 5-5. With a tightened yield requirement, Y > 85% another corresponding solution that satisfies this constraint can be easily picked from that set. The Pareto optimal solutions found using NSGA II with N=20 are listed in Table 5-3. The velocity, porosity and yield objective trend obtained using the NSGA II method is shown in Figure 5-6.

Table 5-3 Pareto optimal set obtained using NSGA II with N=20 and G=16

Pareto Optimal Set (NSGA II N=20 G=16)									
Variables				Objectives					
a_r	b_r	g_r	r_h	V_x (cm/s)	V_y (cm/s)	V_z (cm/s)	Velocity (cm/s)	Porosity (%)	Yield (%)
68.7	10.0	23.1	32.9	46.6	41.9	42.3	75.6	0.0	87.1
58.4	10.0	23.1	63.9	35.2	42.6	43.1	70.1	0.0	86.1
62.3	13.9	26.8	63.9	33.6	39.8	39.5	65.6	0.0	84.4
59.7	12.9	28.3	61.3	39.0	41.1	30.9	64.3	0.0	82.8
54.5	13.9	32.8	63.9	39.0	37.7	20.0	57.8	0.0	79.1
70.0	12.9	34.3	61.3	39.2	35.2	22.4	57.2	0.0	76.8

5.3 Performance Comparison of WS-GA and MOEA

In this study, the success of the WS-GA method depended on the proper objective weights and constraint formulations. Using MOEA, the results obtained were guaranteed to satisfy all the design requirements since constraints are explicitly handled. The feasible optimal solutions obtained in the WS-GA optimization are plotted together with the

MOEA solutions in Figure 5-5. Each point obtained using the WS-GA method shown in Figure 5-5 was found in a separate optimization run. Thus, the four WS-GA points shown on the plot is actually a result of four different optimization runs in total with 1210 design evaluations involved. This large number of evaluations involved is not so desirable for design processes of real casting components.

It is also apparent that the NSGA II technique was able to find a good spread of optimal solutions in just one run. Therefore, it required significantly less number of design evaluations to generate the set of Pareto optimal solutions compared to WS-GA. Using N=20, a total of 320 designs were required to generate the six Pareto optimal points shown in Figure 5-5.

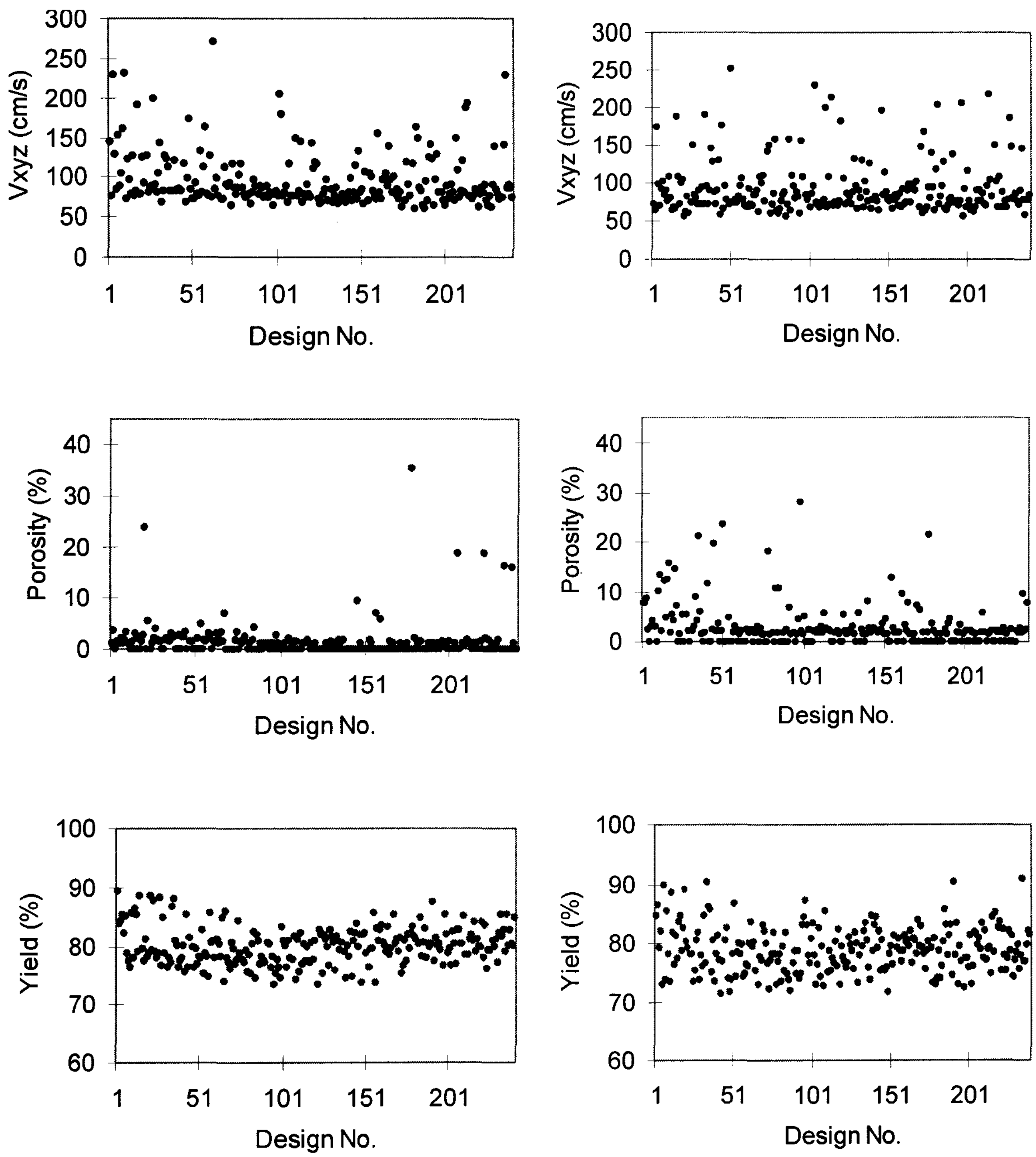
In terms of objective trends, it can be seen from Figures 5-3 and 5-6 that the objective trends were more convergent when WS-GA was used compared to the NSGA II technique. The trend patterns were more obvious in WS-GA as the search narrowed down to the good solutions. Using NSGA II however, the solutions were more diversified and the trend patterns were less evident.

This observation can be attributed with the different performance measures used between the two techniques. In WS-GA, the search is guided by the weighted objective measure and the search converges to one solution as it tried to minimize that single measure. Each objective trend pattern portrays the search relative to the weightings. On the contrary, in MOEA, multiple conflicting objectives have to be taken into consideration simultaneously. Since the search is not guided by just a single measure, the search converges more slowly and therefore has a wider range of variation in the solutions.

For this reason, the ability of MOEA techniques in finding a globally optimal solution is generally slower than the weighted sum method. However, when conflicting objectives are involved, the optimal solution found using the WS-GA approach is largely affected by the choice of weights and constraints as shown earlier. This makes it less reliable for the casting design problem. With the ability to provide reliable results and generate the Pareto front efficiently, the advantages of MOEA considerably outweigh this weakness.

For the gating and riser design optimization problem, comparisons of the MOEA and WS-GA methods can be summarized in the following few aspects:

- *Reliability*: MOEA was more reliable in finding optimal solutions compared to WS-GA because the results are independent of any a priori decision making process. Using WS-GA, incorrect weight settings led to infeasible results even when better solutions existed.
- *Ease of Use*: From the problem formulation standpoint, MOEA was easier to use than WS-GA since it did not require auxiliary knowledge and/or trial and error for weight settings, fitness scaling and constraint formulation.
- *Generality*: Since custom formulations are not required, MOEA is more generalized for the casting design problem compared to WS-GA because it did not need to be modified for each different case.
- *Efficiency*: In terms of generating the Pareto set, MOEA was more efficient as it managed to identify multiple Pareto optimal solutions in a single run compared to WS-GA where it had to be iterated multiple time using different weights.
- *Flexibility*: Using the MOEA technique gave more flexibility in decisions making since trade-off information is given by the Pareto optimal set. This is advantageous from the practical standpoint because objective priorities often change according to current conditions in real world problems.



(a) (b)

Figure 5-6 Velocity, porosity and yield result patterns using N=20 for
(a) WS-GA and (b) NSGA II.

5.4 Optimization with DOE

5.4.1 DOE Benchmark Setup

A benchmark was designed using the Optimal Latin Hypercube (OLH) method. Using this technique, large design spaces can be efficiently sampled and points can be spread evenly within the design space defined by the lower and upper level of each variable [67]. The sampled points are uniformly divided with the same number of divisions, *n* for all factors. These levels are then optimally combined to define the *n* points of the design matrix with each factor level studied only once. With the same number of points, the design space can be more efficiently sampled compared to Orthogonal Arrays, a popular experimental design method [68]. An example of Orthogonal Array and Optimal Latin Hypercube design space sampling with two factors X1 and X2 and nine points is demonstrated in Figure 5-7.

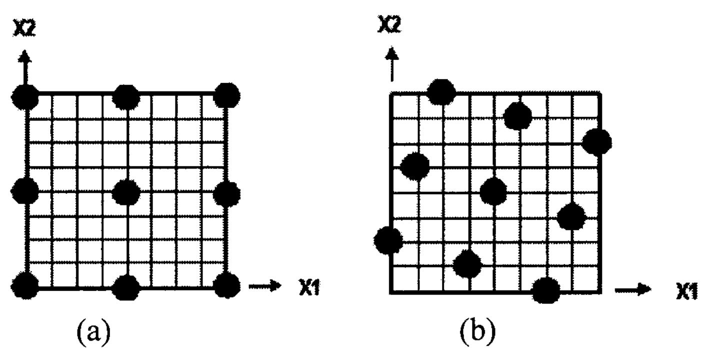


Figure 5-7 Design space sampling using (a) Orthogonal Array and (b) Optimal Latin Hypercube. (adapted from Design of Experiments [69])

Two sets of experiments with 100 (OLH 100) and 200 (OLH 200) points respectively were created to compare the feasibility of using evolutionary algorithms with the popular design-of-experiment (DOE) method. There are four factors which correspond to the four design variables defined in Table 5-1. In OLH 100, each factor is uniformly divided into 100 levels and 100 experimental points are then optimally combined such that each factor level is studied only once. In OLH 200, each factor is divided into 200 levels, allowing more combinations of solutions to be studied.

The ability of WS-GA and MOEA in finding optimal solutions within restricted time limitations or with fewer design evaluations is studied using OLH 100. Since the strength of WS-GA and MOEA lies in the concept of evolution, improvements in the solutions with more time given are studied using OLH 200. The velocity, porosity and metal yield objective values were measured separately for each experimental point. Their results were compared with the solutions found using the MOEA and WS-GA techniques with N=20.

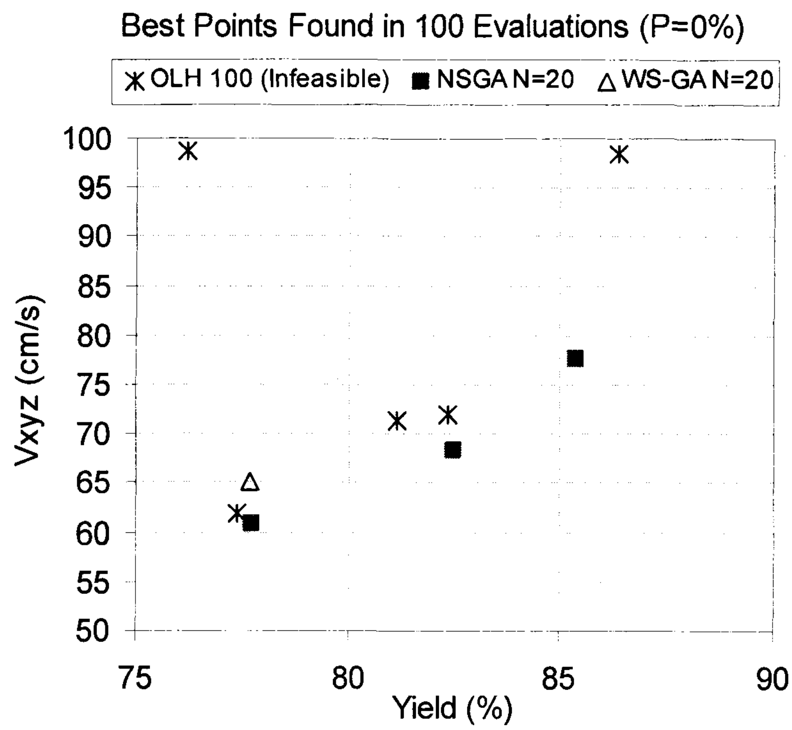
5.4.2 DOE Benchmark Results

With the design requirements of $P = 0\%$, V_x, V_y and $V_z < 50$cm/s and $Y > 85\%$, none of the designs in both sets of experiments OLH 100 and OLH 200 turned out to be feasible. None of the combinations given by the DOE method could optimize all the objectives simultaneously while adhering to the requirement constraints. This shows that even with the optimal combinations of as many as 200 points to sample the design space, feasible or better design points could be missed. For comparison purposes, only selected points in each set of the experiments with $P = 0\%$ are shown in Figure 5-8.

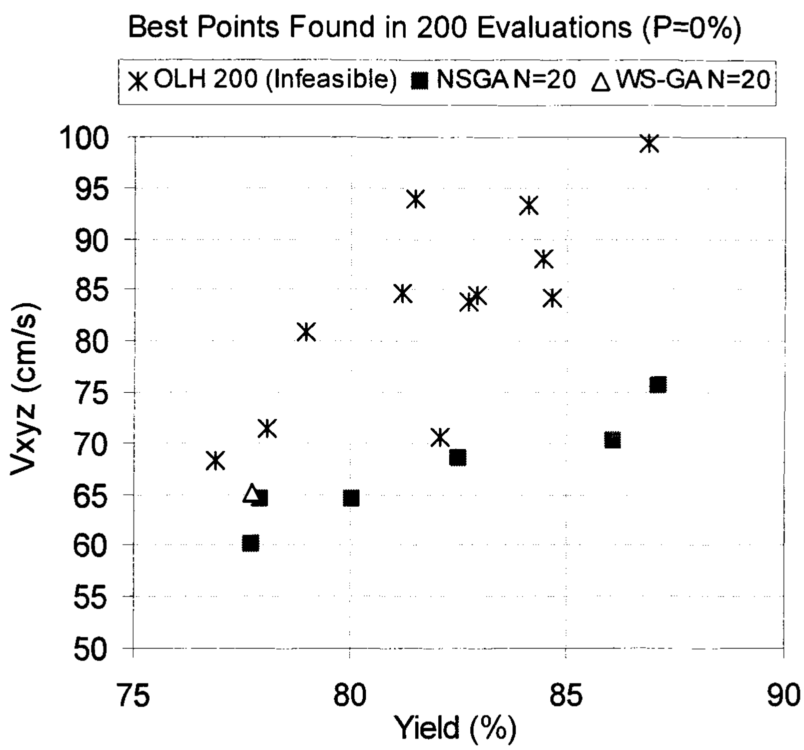
5.4.3 Comparison of DOE and Evolutionary Algorithms

First of all, the WS-GA and MOEA methods were both able to obtain feasible result(s) after 100 and 200 design evaluations whereas the DOE failed to find any. This shows that evolutionary algorithms are more reliable in finding optimal solutions compared to the static DOE method. Within 100 evaluations, the solutions obtained using MOEA were slightly better than the DOE method in terms of the velocity and yield objectives as shown in Figure 5-8(a). One notable achievement is that even with a limited number of designs involved, three feasible designs were successfully found using the MOEA method. In addition, the single optimal solution found using the WS-GA method is dominated by the solution found using MOEA.

With 200 design evaluations, considerable improvements of the solutions were obtained using MOEA as shown in Figure 5-8(a). Six feasible points were successfully found when the DOE still failed to find any. This observation is attributed with the ability of MOEAs to evolve and find better solutions with time. The single solution obtained in WS-GA is comparable to one of the optimal points found using MOEA in terms of non-domination.



(a)



(b)

Figure 5-8 Comparison of best points found using a structured DOE of (a) 100 and (b) 200 designs with WS-GA and NSGA II.

Although the DOE method is simple and popularly used, the results of this study show that evolutionary algorithms have several advantages for the casting design application. Comparisons of the DOE method with evolutionary algorithms (specifically MOEA and WS-GA) from this study can be summarized in the following two main aspects:

- *Reliability*: Optimization using EAs is more reliable than DOE not only in finding feasible designs but also in finding optimal solutions. DOE depends strongly on the pre-defined combinations of points that may miss good solutions whereas EAs are robust search algorithms with meta-heuristic to guide its search.

- *Efficiency*: Optimization using EAs is more efficient than DOE since better solutions were found using EAs both within 200 design evaluations and within a restricted time constraint of 100 evaluations. Furthermore, the MOEA method found multiple Pareto optimal solutions and provided additional trade-off information within the same number of design evaluations.

5.5 Comparisons of Different Designs

Comparisons of some gating and riser designs are shown in Figures 5-9 and 5-10. The effect of using differently sized risers on the shrinkage porosity formed in the casting is shown in Figure 5-9. It can be seen that using top risers with a low height r_h and wide radius a_r could not eliminate the porosity in the casting. Similar infeasible results were obtained when a tall riser with a smaller radius was employed. Porosity from the casting could be eliminated only with an optimally sized riser. The identification of the proper riser size was autonomously examined by the optimization algorithm and did not require any human intervention.

The influence of the runner and ingates sizes on the liquid metal velocity in the casting is shown in Figure 5-10. It can be observed that a narrow gating system with a small radius g_r resulted with higher flow velocity. Using a larger g_r for the gating system increased the gating ratio and allowed the liquid metal to be slowed down before entering the casting cavity.

From the results obtained using both the MOEA and WS-GA methods, it is observed that the optimal gating and riser designs had an a_r in the range of approximately 50 - 62 mm, b_r between 10 - 15 mm, g_r in the 20 - 33 mm range and r_h was approximately 60 mm.

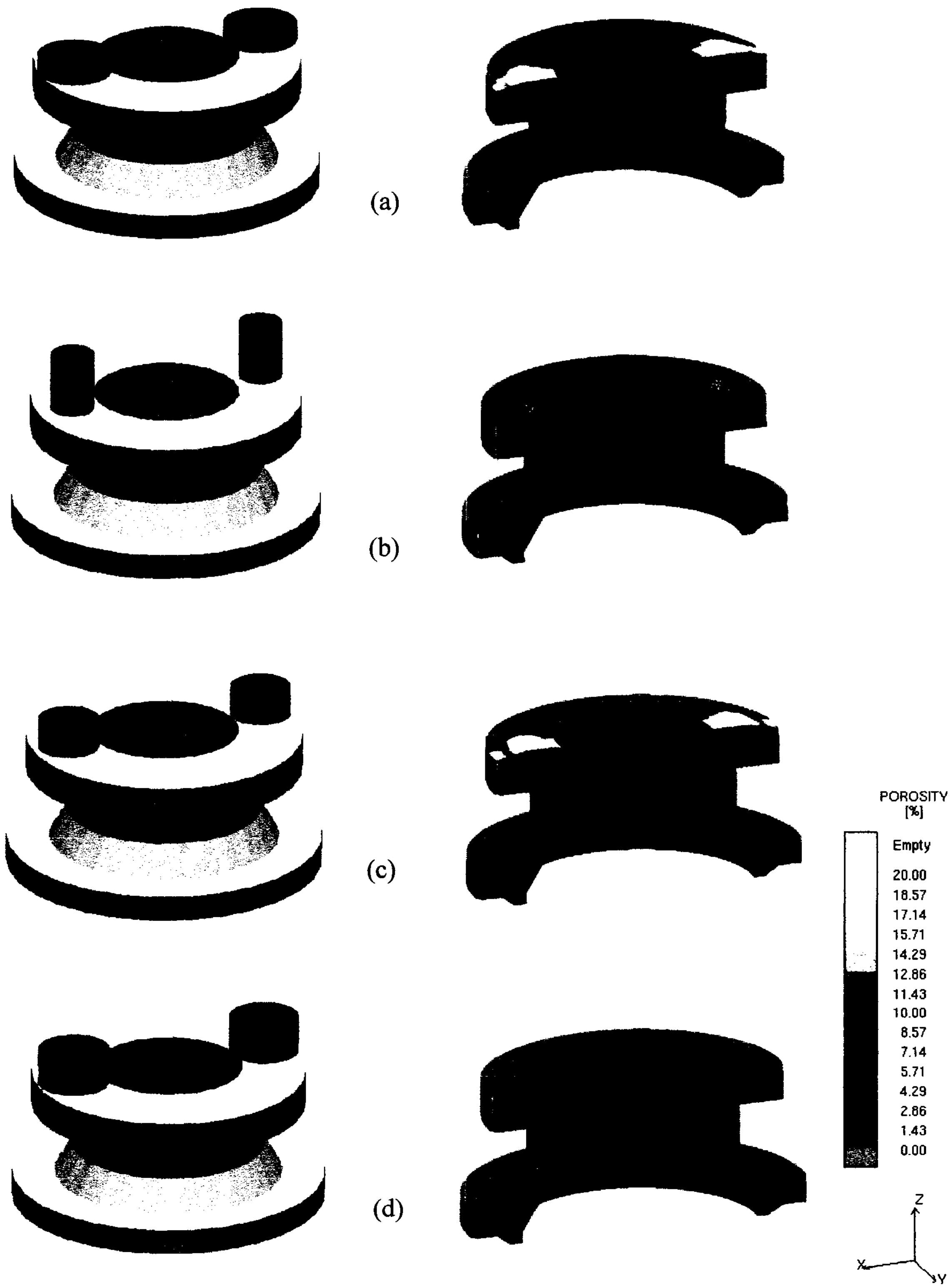


Figure 5-9 Shrinkage porosity results of the casting with a (a) short, wide riser, (b) tall, thin riser, (c) medium sized riser, and (d) an optimal riser.

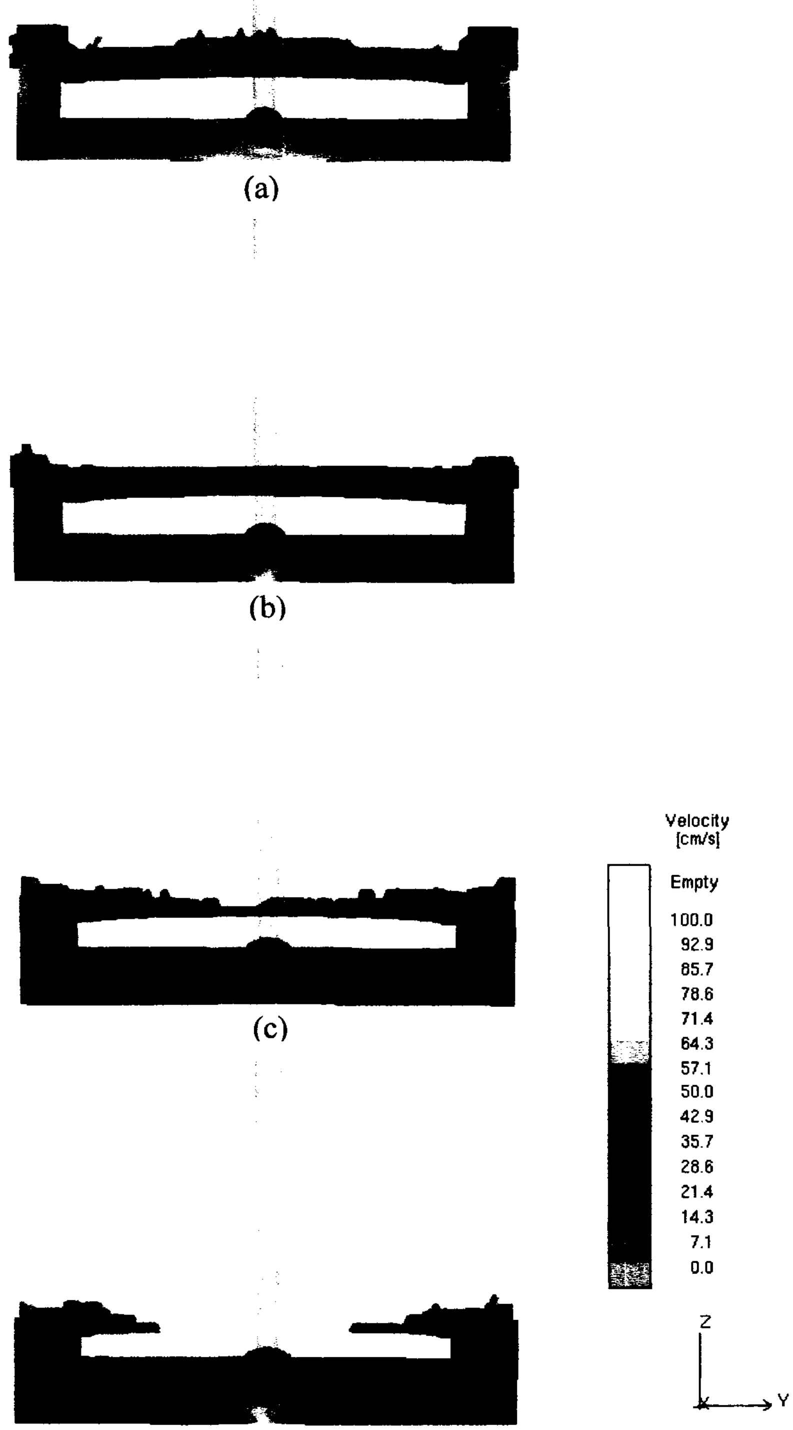


Figure 5-10 Filling velocity results in the casting with gating size of (a) $g_r = 20$ mm, (b) $g_r = 25$ mm, (c) $g_r = 30$ mm, and (d) $g_r = 35$ mm

Chapter 6
Conclusions and Future Work

Numerical simulation allows flexibility in exploring different designs while evolutionary algorithms can evolve and optimize them. Based on the proposed optimization framework, an optimal gating and riser system design could be successfully obtained even with the absence of auxiliary knowledge or analytical information in the problem formulation. Similarly, an initial population made up of all bad designs did not impede the ability of the optimization algorithm in finding better feasible solutions. Thus it can be said that evolutionary optimization techniques are robust even in the complex search space of casting design problem.

For both the WS-GA and MOEA approaches, a population size of 20 was the appropriate size for finding the optimal solution efficiently. The moderately-sized population allowed a sufficient pool of genes to exist in the population for effective crossover operations compared to an inadequate size of 10 or 12. At the same time, it could obtain optimal results effectively without the need for excessive computational resources as with the population size of 40 or 50.

In the scalar optimization approach, WS-GA managed to find good results only when the suitable weights and constraints were properly chosen. In this study, when the yield constraint was changed from 70% to 85%, the weights had to be modified and WS-GA had to be run 4 times using different objective weight settings before a feasible optimal solution could finally be found. The best weight setting that yielded the feasible solution was given in Set #4 (V_{xyz} = 0.25, Y = 0.15 and P = 0.60). Since there is insufficient information for assessing problem-specific parameters in a casting design problem, improper weighting leads to inconsistencies in both the problem formulation and results. Meanwhile, using the vector optimization approach, MOEA only had to be run once regardless of changes in the priorities or requirements and a corresponding feasible design can then be picked from the set of solutions obtained.

The WS-GA method required a total of 1210 design evaluations before the Pareto set of four feasible solutions could be generated. However, the MOEA method managed to obtain six Pareto optimal points that dominated half the solutions found in WS-GA after only 320 design evaluations. This achieved approximately four-fold reduction in the number of design evaluations. The proposed evolutionary optimization approach also proved to be superior in terms of reliability and efficiency compared to the popular DOE method in terms of the number of design evaluations involved and solution optimality.

Overall, the underlying strength of MOEA is the capability to not only provide multiple Pareto optimal solutions but also give trade-off information between the competing yield and quality objectives. This gives an insight into the system characteristics and allows flexibility in decision making. With a clearer understanding of the system, the designer can have a better awareness of the priorities among the objectives before making well-informed decisions.

From this study, the MOEA method proved to be more reliable and efficient than the WS-GA method. It is also easier to use and more generalized for the casting design application. Since the gating and riser design is multi-objective in nature, solving it as a multi-objective optimization problem proved to be the better approach.

The main drawback of using evolutionary techniques with numerical simulation is that it can be computationally expensive as it requires many function evaluations. However, the ever improving computer technology and the option of parallel processing can lead to faster performance.

In this study, the optimization framework with NSGA II managed to autonomously obtain 6 optimal design choices in less than 3 days. Using the trial-and-error approach to manually design, iterate and redesign could possibly take up much more man hours and yet may not necessarily obtain optimal designs.

In casting design optimization, the main areas for enhancements are improving the speed and robustness of the process. Some of the potential future works are as follows:

- Because this proposed optimization framework was tested on a sand casting application, the next step is to apply it on other casting processes such as die casting and different cast alloys. More complicated casting design with more design variables can also be applied to further validate its robustness.

- To improve the speed of the optimization process, parallel processing techniques can be applied to distribute the computational workload. Due to the nature of genetic algorithm that deals with a population of solution in parallel, this technique is also very suitable and straightforward. With proper job allocation and communication of results between processing cores, the reduction of total optimization time can be multi-fold.

- In order to alleviate the computational complexity of lengthy function evaluations, approximation models can be employed to substitute the actual evaluation function. The approximation model can also be trained online using the data generated by the GA in the first few generations and the remaining generations can be evaluated against the model built. This can significantly reduce the evaluation time and yet preserve the results integrity.

References

[1] J. A. Dantzig, S. A. Ebrahimi, and D. A. Tortorelli, "Optimal Casting Design," In Materials Processing: Theory, Methods and Applications", NUMIFORM, Balkema, pp. 1119–1124, 1995.

[2] N. R. Green and J. Campbell, "Influence of Oxide Film Filling Defects on The Strength of Al-7si-Mg Alloy Castings", AFS Transactions, 102:341-347, 1994.

[3] M. Rezvani, X. Yang, and J. Campbell, "The Effect of Ingate Design on The Strength and Reliability of Al. Castings", AFS Transactions, 107:181-188, 1999.

[4] R. Fuoco and E. R. Correa, "The Effect of Gating System Design on The Quality of Aluminum Gravity Castings", Website Article, http://www.moderncasting.com, 2004.

[5] S. Valtierra-Gallardo, J. Talamantes-Silva, and J.A. Gonzalez–Villarreal, "Gating Design After Campbell - A Practical Experience with The Daily Struggle with Porosity", The John Campbell Symposium, The Minerals, Metals & Materials Society, 455-462, 2005.

[6] M Masoumi et al., "Effect of Gating Design on Mold Filling", Transactions of The American Foundry Society, 113:185-196, 2005

[7] S. Guleyupoglu, "Casting Process Design Guidelines", AFS Transactions, 105:869, 1997.

[8] J. Campbell, "The Ten Castings Rules Guidelines For The Reliable Production of Reliable Castings: A Draft Process Specification", Proceedings of The Materials Solutions Conference on Aluminum Casting Technology, Chicago, pp. 3-19, 1998.

[9] J. Campbell, "Castings", Oxford, England, Butterworth-Heinemann Ltd., 1991.

[10] C. E. Esparza, M. P. Guerrero-Mata, R. Z. Ríos-Mercado, "Optimal Design of Gating Systems by Gradient Search Methods", Computational Materials Science, Elsevier, 36(4): 457-467, 2006.

[11] E. C. Fonseca, J. Schmidt, and C. Kleeberg, "Feeding Optimization of A Gray Iron Casting", Metal Asia, 6(6)11-12, 1999.

[12] R.M. Mcdavid and J. Dantzig, "Design Sensitivity and Finite Element Analysis of Free Surface Flows with Application to Optimal Design of Casting Rigging Systems", International Journal For Numerical Methods in Fluids, Wiley Interscience, 28(3):419 - 442, 1998.

[13] Y. Otsuka, G. Trapaga and J. Szekely, "Optimal Casting Design by Application of Mold Filling and Solidification Simulation", Light Metals, Minerals, Metals and Materials Society, USA, 897-905, 1994.

[14] C.C Tai and J.C. Lin, "A Runner-Optimization Design Study of A Die-Casting Die", Journal of Materials Processing Technology, 84(1):1-12, 1998.

[15] M. J. Marques, "CAE Techniques For Casting Optimization", Website Article from CALCOM ESI, http://www.calcom.ch.

[16] D. C. Schmidt, J. F. Meredith, "One Step Beyond: From Simulation to Optimization", Metal Casting Technologies, 47(2):22-24, 2001.

[17] R.M. McDavid and J. Dantzig, "Fluid Flow in Casting Rigging Systems: Modeling, Validation, and Optimal Design", Metallurgical and Materials Transactions, USA, 1998.

[18] S. A. Ebrahimi, D. A. Tortorelli and J. A. Dantzig "Sensitivity Analysis and Nonlinear Programming Applied to Investment Casting Design", Applied Mathematical Modelling, 20(11):792-799, 1996.

[19] B. Ravi, "Feeder Design and Solidification Analysis", Online Lecture Notes, Mechanical Engineering Department, Indian Institute of Technology, Powai, Mumbai.

[20] S. Das, G. Upadhya and A. J. Paul, "A Geometric Approach to Optimized Riser Design in Castings", The Minerals, Metals & Materials Society, USA, pp. 955-965, 1994.

[21] S. Guleyupoglu et al, "Casting Riser Design Optimization Using Genetic Algorithms", Modeling of Casting, Welding and Advanced Solidification Processes VII, London, 10-15: 389-396, 1995.

[22] Jacob et al, "Intelligent Design of Feeders For Castings by Augmenting CAD with Genetic Algorithms", Journal of Intelligent Manufacturing, Springer, Netherlands, 15(3):299-305, 2004.

[23] J. Andersson, "Multiobjective Optimization in Engineering Design: Applications to Fluid Power Systems", M. Sc. Thesis, Linköpings University, Sweden, 2001.

[24] S. Obayashi et al, "Multiobjective Evolutionary Computation For Supersonic Wing-Shape Optimization", Evolutionary Computation, IEEE Transactions, 4(2):182-187, 2000.

[25] J. Andersson, "A Survey of Multiobjective Optimization in Engineering Design", Technical Report LiTH-IKP-R-1097, Department of Mechanical Engineering, Linköping University, Linköping, Sweden, 2000.

[26] C. A. Coello, "Collections on Evolutionary Multiobjective Optimization", Website Article, http://delta.cs.cinvestav.mx/~ccoello/emoo/emoochapters.html, Last updated on October 12, 2006.

[27] O.T. Pham and D. Karaboga, "Intelligent Optimization Techniques: Genetica Agorithms, Tabu Search, Simulated Annealing and Neural-Networks", Springer-Verlag, 2000.

[28] D. F. Jones, S. K. Mirrazavi and M. Tamiz , "Multi-Objective Meta-Heuristics: An Overview of The Current State-of-The-Art", European Journal of Operational Research, Elsevier, 137(1):1-9, 2002.

[29] J. Campbell and R.A. Harding, "The Filling of Castings", TALAT Lecture 3203, European Aluminium Association, 1994.

[30] J. Runyoro, S. M. A. Boutorabi, and J. Campbell, "Critical Gate Velocities For Film-Forming Castings Alloys: A Basis For Process Specification", AFS Transactions, 100:225-34, 1992.

[31] J. Campbell, "The Ten Castings Rules Guidelines For The Reliable Production of Reliable Castings: A Draft Process Specification", Proceedings of the Materials Solutions Conference on Aluminum Casting Technology, Chicago, pp. 3-19, 1998.

[32] J. A. Arora, "Introduction to Optimum Design", 2nd Edition, Elsevier Academic Press, 2004.

[33] S. S. Rao, "Engineering Optimization Theory and Practice", 3rd Edition, Wiley Interscience, 1996.

[34] C.M. Fonseca and P.J. Fleming, "Genetic Algorithms For Multi-Objective Optimization: Formulation, Discussion and Generalization", Proceedings of The Fifth International Conference, Morgan Kaufmann, San Mateo, CA, pp. 141-153, 1993.

[35] C. A. Coello Coello, "An Updated Survey of GA-Based Multiobjective Optimization Techniques", ACM Computing Surveys, 32(2):109-143, 2000.

[36] E. Zitzler, M. Laumanns and S. Bleuler, "A Tutorial on Evolutionary Multiobjective Optimization", Workshop on Multiple Objective Metaheuristics, Springer-Verlag, Berlin, Germany, 2004.

[37] A. Marczyk, "Genetic Algorithms and Evolutionary Computation", Website Article, http://www.talkorigins.org, Last updated on April 23, 2004.

[38] R. Haupt and S. E. Haupt, "Practical Genetic Algorithms", John Wiley & Sons, 1998.

[39] J. H. Holland, "Adaptation in Natural and Artificial Systems", Ann Arbor: The University of Michigan Press, 1975.

[40] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, 1989.

[41] D. E. Goldberg and K. Deb, "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms", In G. Rawlins, Ed., Foundations of Genetic Algorithms, Morgan Kaufmann, 1991.

[42] K. De Jong, "An Analysis of The Behavior of A Class of Genetic Adaptive Systems", PhD Thesis, University of Michigan, Dissertation Abstracts International, 36(10):5140b, University Microfilms No. 76-9381, 1975.

[43] D. Thierens and D. Goldberg, "Mixing in Genetic Algorithms", Proceedings of The Fifth International Conference on Genetic Algorithms, San Mateo, CA, Morgan Kaufmann, pp. 38-45, 1993.

[44] C. T Lin. C. S. G. Lee , "Genetic Algorithms", A Neuro-Fuzzy Synergism to Intelligent System, Prentice Hall, pp.382 – 413, 1996.

[45] L. Booker, "Improving Search in Genetic Algorithms", Genetic Algorithms and Simulated Annealing, Morgan Kaufmann Publishers, 1987.

[46] W. M. Spears and K. A. De Jong, "An Analysis of Multi-Point Crossover", Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, G. J. E. Rawlins, pp. 301-315, 1991.

[47] J. D. Schaffer et al., "A Study of Control Parameters Affecting Online Performance of Genetic Algorithms For Function Optimization", Proceedings of The 3rd International Conference on Genetic Algorithms, USA, pp. 51-60, 1989.

[48] J. Grefenstette, "Optimization of Control Parameters for Genetic Algorithms", IEEE Transactions on Systems, Man and Cybernetics, pp. 122 – 128, 1986.

[49] P. Reed et al. , "Designing A Competent Simple Genetic Algorithm For Search and Optimization", Water Resources Research, 36(12):3757-3762, 2000.

[50] J. Hessner and R. Manner, "Choosing Optimal Mutation Rate", In Proceedings of The First Workshop on Parallel Problem Solving From Nature, Berlin: Springer-Verlag, pp. 23-31, 1991

[51] B. Minsker and P. Reed, "Genetic Algorithms For Effective Solution of Real-World Optimization Problems", Presentation at the ASCE World Water and Environmental Resources Conference in Philadelphia, PA, 2003.

[52] F. Zhang, S. Chen & M. Khalid, "Multi-Point Optimization of Transonic Wing by Real-Coded Genetic Algorithm", Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, 2004.

[53] D. L. Carroll, "FORTRAN Genetic Algorithm Driver", Website Article, University of Illinois, http://www.staff. uiuc.edu/carroll/ga.html, 1997.

[54] C. A. Coello Coello and G. T. Pulido, "A Micro-Genetic Algorithms For Multiobjective Optimization", Proceedings of The 1st International Conference of Evolutionary Multi-Criterion Optimization, Zurich, Switzerland, Springer Berlin Heidelberg, 1993/2001:126-141, 2001.

[55] D. E Goldberg, K. Deb and J. H. Clark, "Genetic Algorithms, Noise, and The Sizing of Populations", Complex Systems, 6:333-362, 1992.

[56] G. Harik et al, "The Gambler's Ruin Problem, Genetic Algorithms, and The Sizing of Populations", Proceedings of IEEE International Conference on Evolutionary Computation, Indianapolis, USA, pp. 7-12, 1997.

[57] P. Reed, "Striking the Balance: Long-Term Groundwater Monitoring Design for Multiple Conflicting Objectives", Ph. D. Thesis, University of Illinois, 2002.

[58] D. Powell and M. M. Skolnick, "Using Genetic Algorithms in Engineering Design Optimization with Non-Linear Constraints", Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 424-430, 1993.

[59] I. Das and J. E. Dennis, "Closer Look At Drawbacks of Minimizing Weighted Sums of Objectives For Pareto Set Generation in Multicriteria Optimization Problems", Structural Optimization, 14:63-69, 1997.

[60] K. Deb et. al., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", Evolutionary Computation, IEEE Transactions, 6(2):182 – 197, 2002.

[61] M. Laumanns, E. Zitzler, and L.Thiele, "On The Effects of Archiving, Elitism, and Density Based Selection in Evolutionary Multi-Objective Optimization", Proceeding of the 1st International Conference on Evolutionary Multi-Criterion Optimization, pp. 181-196, 2001.

[62] D. W. Corne, J. D. Knowles, M. J. Oates, "Pareto Envelope-Based Selection Algorithm", Proceedings of the Parallel Problem Solving from Nature VI Conference, pp. 839-848, 2000.

[63] N. Srinivas, and K. Deb, "Multi-Objective Function Optimization Using Non-Dominated Sorting Genetic Algorithms", Evolutionary Computation, 2(3):221–248, 1995.

[64] J. Horn, N. Nafploitis, and D. E. Goldberg, “A Niched Pareto Genetic Algorithm For Multi-Objective Optimization”, Proceedings of The First IEEE Conference on Evolutionary Computation, pp. 82–87, 1994.

[65] D. Kunkle, “A Summary and Comparison of MOEA Algorithms”, Technical Report, Computer and Information Science, Northeastern University, Boston, Massachusetts, 2005.

[66] E. K. Antonsson and J. Cagan, “Formal Engineering Design Synthesis”, Cambridge University Press, 2001.

[67] K. Q. Ye, W. Li, and A. Sudjianto, “Algorithmic Construction of Optimal Symmetric Latin Hypercube Designs”, Journal of Statistical Planning and Inference, pp. 145-159, 2000.

[68] A.B. Owen, “Orthogonal Arrays for Computer Experiments, Integration and Visualisation”, Statistica Sinica 2, pp. 439-452, 1992.

[69] iSIGHT Reference Guide, Version 9.0, Engineous Software Inc., United States, 2004.

[70] C. M. Fonseca and P. J. Fleming, “Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms Part I: A Unified Formulation” IEEE Transactions on Systems, Man and Cybernetics, 28(1):26-37, 1998.

[71] M. Laumanns, E. Zitzler, and L.Thiele, “A Unified Model For Multi-Objective Evolutionary Algorithms with Elitism”, Congress on Evolutionary Computation, Piscataway, New Jersey, 1:46-53, 2000.

[72] H. Eschenauer, J. Koski and A. Osyczka, “Multicriteria Design Optimization: Procedures and Application”, Springer-Verlag, 1990

[73] J. Kor et. al., “Numerical Understanding of John Campbell's Casting Design Rules”, Simulation of Aluminum Shape Casting Processing: From Alloy Design to Mechanical Properties, TMS, pp. 177-187, 2006.

[74] MAGMASOFT® Reference Guide, Version 4.2, MAGMA GmbH, Aachen, Germany, 2002.

Vita Auctoris

Name:	Jean Shang Leen Kor
Place of Birth:	Kuala Lumpur, Malaysia
Year of Birth:	1981
Education:	Convent High School, Kajang, Malaysia 1994 -1999
	University of Windsor, Windsor, Ontario 2000 - 2004 B.A.Sc.
	University of Windsor, Windsor, Ontario 2005 - 2006 M.A.Sc.
Co-op Work Experience:	Ford Engine Plant, Windsor, Ontario Summer 2001
	Ford Casting Plant, Windsor, Ontario Summer 2002 and 2004
	Kautex Textron, Windsor, Ontario Winter 2003